SIMULIA

3DEXPERIENCE

ABAQUS 2026 FD04
VERIFICATION GUIDE

DASSAULT SYSTEMES

Contents

Trademarks and Legal Notices....16
Abaqus Verification Guide....17
What's New....18
Analysis Procedures and Techniques Verification....37
Dynamic analysis....38
Modal dynamic analysis with baseline correction....39
Steady-state dynamic analysis for two-dimensional elements....41
Steady-state dynamic analysis for infinite elements....44
Random response analysis....46
Single degree of freedom spring-mass systems....48
Linear kinematics element tests....50
Mass scaling....53
Crack propagation....60
Crack propagation analysis....61
Propagation of hydraulically driven fracture....67
Hydraulically driven branching fracture....68
Modeling slurry transport and placement in hydraulically driven fracture....69
Substructuring....72
Substructure rotation, mirroring, transformation, and constraints....73
Substructure recovery with The nodal Transformation....79
Degenerated elements within a substructure....80
Substructure load case with centrifugal loads....82
Substructure load case with nodal pressure loads....83
Thermal-stress analysis with substructures....84
Substructure preload history....86
Substructure removal....89
Substructure damping....90
Substructures with rebar....91
Frequency extraction for substructures....93
Substructures with large rotations....94
Coupled structural-acoustic analysis with substructures....101
Additive manufacturing process simulation....104
Heat energy balance....105
Pattern-based eigenstrain analysis....110
Trajectory-based eigenstrain analysis....116
Metallurgical phase transformations....120
Material deposition....128
Moving heat sources....130
Material removal....132
Electromagnetic analysis....134
Eddy current analysis....135
Magnetostatic analysis....138
Coupled thermal-electrochemical analysis....140

Fully coupled thermal-electrochemical procedures....141
Modeling a half-cell lithium metal anode battery with multiphysics Butler-Volmer surface loading....145
Modeling a solid electrolyte and solid-state battery....148
Piezoelectric analysis....150
Static analysis for piezoelectric materials....151
Frequency extraction analysis for piezoelectric materials....153
General analysis procedures for piezoelectric materials....154
Submodeling....159
General submodeling....161
Two-dimensional continuum stress/displacement submodeling....170
Three-dimensional continuum stress/displacement submodeling....174
Cylindrical continuum stress/displacement submodeling....177
Axisymmetric continuum stress/displacement submodeling....179
Axisymmetric stress/displacement submodeling with twist....181
Membrane submodeling....183
Shell submodeling....185
Surface element submodeling....192
Heat transfer submodeling....194
Coupled temperature-displacement submodeling....198
Pore pressure submodeling....206
Piezoelectric submodeling....211
Acoustic submodeling....214
Shell-to-solid submodeling....217
Gasket submodeling....224
Miscellaneous submodeling tests....226
Acoustic and shock analyses....231
Volumetric drag....232
Impedance boundary conditions....234
Perfectly matched layers....241
Transient acoustic wave propagation....242
Adaptive meshing applied to coupled structural-acoustic problems....245
CONWEP blast loading pressures....251
Blast loading of a circular plate using the CONWEP model....254
Model change....255
Stress/displacement model change: static....256
Stress/displacement model change: dynamic....262
Stress/displacement model change: general tests....265
Heat transfer model change: steady state....270
Coupled temperature-displacement model change: steady state....272
Contact model change....275
Acoustic model change: steady state....276
Pore-thermal model change....278
Symmetric model generation and analysis of cyclic symmetry models....280
Symmetric model generation and results transfer....281
Analysis of cyclic symmetric models....296
Abaqus/Aqua analysis....303
Aqua load cases....304
Jack-up foundation analysis....313

Elastic-plastic joint elements....319
Design sensitivity analysis....321
Direct design sensitivity analysis....322
Adjoint design sensitivity analysis....332
Transferring results between Abaqus/Standard and Abaqus/Explicit....339
Transferring results between Abaqus/Explicit and Abaqus/Standard....340
Transferring results from one Abaqus/Standard analysis to another Abaqus/Standard analysis....346
Transferring results from one Abaqus/Explicit analysis to another Abaqus/Explicit analysis....354
Transferring results with general beam sections....361
Transferring results with general shell sections....362
Adding and removing elements during results transfer....366
Transferring rigid elements....371
Transferring mass and rotary inertia elements....374
Transferring connector elements into Abaqus/Explicit....377
Transferring hourglass forces....380
Changing the material definition during import....383
Transferring results with plasticity....385
Transferring results with damage....394
Transferring results with hyperelasticity....400
Transferring results with viscoelasticity....404
Transferring results for a hyperelastic sheet with a circular hole....407
Transferring results with hyperfoam....409
Transferring results with orientation....412
Miscellaneous results transfer tests....419
Transferring results with parallel rheological framework....440
Transferring results with nodal temperature and field variables....442
Transferring results of an analysis model multiple times....447
Transferring results from multiple previous analyses....455
Meshed beam cross-sections....460
About meshed beam cross-sections....461
Meshing and analyzing a two-dimensional model of a beam cross-section....462
Using generated cross-section properties in a beam analysis....465
Co-simulation....467
Fluid-structure interaction of a cantilever beam inside a channel....468
Abaqus/Standard to Abaqus/Explicit co-simulation....477
Adaptive remeshing....484
Pressurized thick-walled cylinder....485
Error indicators....488
Discrete particle methods....493
Discrete element method analysis....494
Discrete particle swelling....504
Particle generator....506
Lumped kinetic molecular method....512
Miscellaneous procedures and techniques....515
Direct cyclic and low-cycle fatigue analyses....516
Complex eigenvalue extraction....519
CEL analysis of a rotating water disk....522
Smoothed particle hydrodynamic analysis....527

Frequency extraction using the AMS eigensolver....540
Steady-state dynamics with nondiagonal damping using the AMS eigensolver....547
Media transport....552
Transferring results between dissimilar meshes in Abaqus/Standard....554
One-step inverse analysis....560
Ramping of initial stress....562
Undrained pore fluid flow and stress analysis....564
Controlling material rate dependence....566
Constraints Verification....575
Boundary conditions....576
Complex boundary conditions....576
Displacement, velocity, and acceleration boundary conditions....577
Velocity-type boundary conditions, static analysis....578
Multi-point constraints....580
LINEARMPC....580
QUADRATIC, BILINEAR, and C BIQUADMPCs....581
P LINEARMPC....583
T LINEARMPC....585
P BILINEARMPC....587
T BILINEARMPC....588
BEAMMPC....590
ELBOWMPC....592
LINKMPC....594
PINMPC....596
REVOLUTEMPC....597
SLIDERMPC....599
UNIVERSALMPC....602
V LOCALMPC....603
SS LINEAR and SLIDERMPCs....604
SS BILINEAR, SSF BILINEAR, and SLIDERMPCs....608
TIEMPC....610
CYCLSYMMPC....613
Internal MPC types BEAMRIGID and BEAMTIE with transforms....615
MPC sorting....617
Surface-based tie constraint....618
Elements tested....618
Features tested....618
Problem description....618
Results and discussion....618
Input files....619
Rigid bodies....624
Rigid body mass properties....625
Tie and pin node sets....629
Rigid body as an MPC....632
Rigid body constraint....635
Including deformable element types in a rigid body....638
Coupling constraints....641
Kinematic coupling constraints....641

Kinematic coupling constraints with local coordinate systems....642
Internal sorting of kinematic coupling constraints....643
Distributing coupling constraints with user-specified weights....644
Default distributing weight factors....646
Weighting method and influence region....647
Collinear coupling node arrangement....648
Moment release for distributing coupling....649
Dimensional coupling....650
Structural coupling....651
Auto convert single facet distributing coupling to tie....653
Rotational coupling for distributing coupling constraint for collinear cloud nodes....654
Kinematic coupling constraint....657
Elements tested....657
Features tested....657
Problem description....657
Results and discussion....658
Input files....658
Shell-to-solid coupling constraints....659
Static tests with shell-to-solid coupling constraints....659
Dynamic tests with shell-to-solid coupling constraints....662
Free vibration of a cantilevered thin square with shell-to-solid coupling constraints....663
Static test of a built-up beam with shell-to-solid coupling constraints....665
Explicit dynamic test of a built-up beam with shell-to-solid coupling constraints....666
Coupling of acoustic and structural elements....668
Elements tested....668
Problem description....668
Results and discussion....669
Input files....669
Mesh-independent spot welds....670
Spot-welded plates subject to pressure and shear loading....670
Multi-layer spot welds between surfaces defined on various element types....671
Single-layer spot welds between surfaces defined on various element types with varying mesh densities.672
Large deformation of a spot-welded beam....673
Spot welds used in various analysis techniques....674
Spot weld surfaces forming T-intersections....674
Linear dynamics....675
Compounded local coordinate systems....675
Embedded element technique....677
Elements tested....677
Features tested....677
Problem description....677
Results and discussion....677
Input files....677
Figures....679
Miscellaneous....680
Reset overconstraint checking controls....681
Thermal expansion of constraints....686
Elements Verification....687

About element verification tests....688
Eigenvalue tests....688
Simple load tests....688
Element loading options....688
Patch tests....689
Contact tests....689
Interface tests....689
Rigid body verification....689
Connector element verification....689
Special-purpose stress/displacement elements....689
Miscellaneous tests....690
Eigenvalue tests....691
Eigenvalue extraction for single unconstrained elements....692
Eigenvalue extraction for unconstrained patches of elements....709
Acoustic modes....713
Simple load tests....716
Membrane loading of plane stress, plane strain, membrane, and shell elements....718
Generalized plane strain elements with relative motion of bounding planes....722
Three-dimensional solid elements....725
Axisymmetric solid elements....728
Axisymmetric solid elements with twist....731
Cylindrical elements....733
Loading of piezoelectric elements....739
Love-Kirchhoff beams and shells....745
Shear flexible beams and shells: I....747
Shear flexible beams and shells: II....750
Initial curvature of beams and shells....753
Normal definitions of beams and shells....758
Constant curvature test for shells....761
Verification of section forces for shells....763
Composite shell sections....765
Cantilever sandwich beam: shear flexible shells....771
Thermal stress in a cylindrical shell....773
Variable thickness shells and membranes....776
Shell offset....780
Axisymmetric membrane elements....782
Cylindrical membrane elements....785
Verification of beam elements and section types....789
Beam added inertia....805
Beam fluid inertia....808
Beam with end moment....811
Flexure of a deep beam....814
Simple tests of beam kinematics....822
Tensile test....829
Simple shear....832
Verification of the elastic behavior of frame elements....834
Verification of the plastic behavior of frame elements....841
Three-bar truss....843

Pure bending of a cylinder: CAXA elements....845
Cylinder subjected to an asymmetric temperature field: CAXA elements....851
Cylinder subjected to asymmetric pressure loads: CAXA elements....856
Cylinder subjected to an asymmetric pore pressure field: CAXA elements....862
Modal dynamic and transient dynamic analysis with CAXA and SAXA elements....867
Simple load tests for thermal-electrical elements....872
Hydrostatic fluid elements....874
Fluid link element....878
Fluid pipe element....882
Thermal fluid pipe elements....884
Temperature-dependent film condition....887
Surface-based pressure penetration....889
Gasket behavior verification....897
Gasket element assembly....903
Cohesive elements....905
Coriolis loading for direct-solution steady-state dynamic analysis....916
Pipe-soil interaction elements....918
Shear panel elements....921
Element loading options....922
Continuum stress/displacement elements....923
Beam stress/displacement elements....953
Pipe stress/displacement elements....963
Shell, membrane, and truss stress/displacement elements....967
Cohesive element load verification....978
ELBOW elements....981
Continuum pore pressure elements....984
Continuum and shell heat transfer elements....1000
Coupled temperature-displacement elements....1006
Coupled thermal-electrical-structural elements....1022
Piezoelectric elements....1026
Continuum mass diffusion elements....1033
Thermal-electrical elements....1035
Rigid elements....1039
Mass and rotary inertia elements....1041
Abaqus/Explicit element loading verification....1042
Surface tension loads....1047
Incident wave loading....1059
Distributed traction and edge loads....1078
Patch tests....1083
Membrane patch test....1084
Patch test for three-dimensional solid elements....1089
Patch test for cylindrical elements....1093
Patch test for axisymmetric elements....1095
Patch test for axisymmetric elements with twist....1098
Patch test for plate bending....1102
Patch test for beam elements....1104
Patch test for heat transfer elements....1107
Patch test for thermal-electrical elements....1110

Patch test for acoustic elements....1112
Connector element verification....1114
Damped free vibration with initial conditions....1115
Sinusoidal excitation of a damped spring-mass system....1117
Multiple instances of connector elements....1119
Individual connector option tests....1126
Connector elements in perturbation analyses....1137
Tests for special-purpose connectors....1142
Special-purpose stress/displacement elements....1147
Flexible joint element....1148
Line spring elements....1153
Distributing coupling elements....1158
Drag chain elements....1164
Miscellaneous tests....1165
Rebar in Abaqus/Standard....1166
Rebar in Abaqus/Explicit....1179
Beam rebar line reinforcement in Abaqus/Standard and Abaqus/Explicit....1183
Convection elements: transport of a temperature pulse....1189
Continuum shells: basic element modes....1190
Transverse shear for shear-flexible shells....1191
Linear dynamic analysis with fluid link....1194
Rigid bodies with temperature DOFs, heat capacitance, and nodal-based thermal loads....1197
Analysis of unbounded acoustic regions....1206
One-dimensional steady-state dynamic solutions for poroelastic acoustic elements....1209
Nonstructural mass verification....1213
Mass adjust verification....1217
Defining a reference mesh....1219
Bending stabilization for full-integration linear elements....1221
Interactions Verification....1223
Small engineering tests with contact....1224
Rolling of steel plate....1225
Beam impact on cylinder....1230
Contact properties....1237
Anisotropic friction for Abaqus/Explicit....1238
Thermal surface interaction....1244
Coupled thermal-electrical surface interaction....1248
Friction models in Abaqus/Standard....1250
Friction models in Abaqus/Explicit....1254
Cohesive surface interaction....1258
Modified contact pressure-overclosure relationship....1267
Interference fit and specified clearance....1269
Contact with time-dependent prescribed interference values....1270
Contact with initial overclosure of curved surfaces....1273
Small-sliding contact with specified clearance or overclosure values....1275
Contact formulations....1280
Small-sliding contact between stress/displacement elements....1281
Small-sliding contact between coupled temperature-displacement surfaces....1287
Small-sliding contact between coupled thermal-electrical-structural surfaces....1293

Small-sliding contact between coupled pore pressure-displacement elements....1297
Finite-sliding contact between stress/displacement elements....1302
Finite-sliding contact between a deformable body and a rigid surface....1305
Finite-sliding contact between a deformable body and a meshed rigid surface....1308
Finite-sliding contact between coupled temperature-displacement elements....1311
Finite-sliding contact between coupled thermal-electrical-structural elements....1320
Finite-sliding contact between coupled pore pressure-displacement elements....1325
Finite sliding between concentric cylinders—axisymmetric and CAXA models....1329
Self-contact of finite-sliding deformable surfaces....1332
Contact searching for analytical rigid surfaces....1335
Surface-to-surface approach for finite-sliding contact....1337
General contact in Abaqus/Standard....1339
Surface smoothing for surface-to-surface contact....1344
Features tested....1344
Problem description....1344
Results and discussion....1344
Input files....1344
Initial contact stress....1345
Elements tested....1345
Problem description....1345
Results and discussion....1345
Input files....1345
Contact wear modeling....1347
Elements tested....1347
Problem description....1347
Results and discussion....1347
Input files....1348
Contact surface representation of threads....1351
Features tested....1351
Problem description....1351
Results and discussion....1351
Input files....1351
Miscellanous....1352
Contact between discrete points....1353
Automatic element conversion for surface contact....1356
Automatic surface definition and surface trimming....1357
Contact surface extensions....1362
Adjusting contact surface normals at symmetry planes....1364
Contact controls....1366
Multiple surface contact with penalty method....1370
Automated contact patch algorithm for finite-sliding deformable surfaces....1375
Materials Verification....1377
Mechanical properties....1378
Elastic materials....1380
Viscoelastic materials....1411
Mullins effect and permanent set....1426
Hysteretic materials....1438
Temperature-dependent elastic materials....1442

Field-variable-dependent elastic materials....1448
Large-strain viscoelasticity with hyperelasticity....1454
Nonlinear large-strain viscoelasticity with hyperelasticity....1457
Transient internal pressure loading of a viscoelastic cylinder....1464
Rate-independent plasticity....1470
Rate-dependent plasticity in Abaqus/Standard....1508
Rate-dependent plasticity in Abaqus/Explicit....1515
Annealing temperature....1519
Temperature-dependent inelastic materials....1521
Field-variable-dependent inelastic materials....1528
Barlat anisotropic plasticity....1535
Johnson-Cook plasticity....1545
Porous metal plasticity....1549
Drucker-Prager plasticity....1553
Drucker-Prager/Cap plasticity model....1559
Paperboard material model....1564
Equation of state material....1568
Progressive damage and failure of ductile metals....1581
Progressive damage and failure in fiber-reinforced materials....1593
Generalized damage and failure for anisotropic materials....1619
Creep....1621
Concrete smeared cracking....1634
Concrete damaged plasticity....1636
Two-layer viscoplasticity....1640
Brittle cracking constitutive model....1642
Cracking model: tension shear test....1647
Hydrostatic fluid....1653
Composite, mass proportional, and rotary inertia proportional damping in Abaqus/Standard....1655
Material damping....1657
Mass proportional damping....1661
Structural damping....1664
Thermal expansion test in Abaqus/Standard....1667
Thermal expansion test in Abaqus/Explicit....1669
Field expansion test....1680
Multiscale materials....1682
Cure modeling in Abaqus/Standard....1712
Plasticity corrections....1718
Electrical and mechanical modeling of cardiac tissue....1720
Thermal properties....1732
Field-variable-dependent conductivity....1733
Conductivity and specific heat....1736
Gap conductance....1737
Piezoresistivity properties: Piezoresistive effect....1739
Elements tested....1739
Problem description....1739
Results and discussion....1739
Input files....1739
Output Verification....1740

Writing element matrix output records....1741
Elements tested....1741
Features tested....1741
Problem description....1741
Results and discussion....1741
Input files....1742
Substructure matrix output request....1744
Elements tested....1744
Features tested....1744
Problem description....1744
Results and discussion....1744
Input files....1744
Integrated output variables....1746
Elements tested....1746
Features tested....1746
Problem description....1746
Results and discussion....1746
Input files....1746
Rigid body motion output variables....1748
Elements tested....1748
Features tested....1748
Problem description....1748
Results and discussion....1749
Input files....1749
Element nodal forces in beam section orientation....1751
Elements tested....1751
Features tested....1751
Problem description....1751
Results and discussion....1751
Input files....1751
User Subroutines Verification....1752
About user subroutine verification tests....1753
User subroutine verification....1754
DFLUX....1756
DISP....1758
DLOAD....1760
FRIC....1761
FRIC_COEF....1765
GAPCON....1769
GAPELECTR....1771
HARDINI....1772
HETVAL....1773
RSURFU....1774
SDVINI....1776
UAMP....1777
UANISOHYPER_INV and VUANISOHYPER_INV....1779
UCOHESIVEOFFSET....1782
UCREEPNETWORK....1784

UCURE....1785
UEL....1788
UELMAT....1792
UEPACTIVATIONVOL....1794
UEXPAN....1795
UFLUID....1796
UGENS....1798
UHARD....1800
UHYPER_STRETCH....1802
UINTER....1806
UMAT and UHYPER....1808
UMATHT....1814
URDFIL....1815
USDFLD....1817
UTEMP, VUTEMP, UFIELD, UMASFL, and UPRESS....1819
UTRSNETWORK....1825
UVAREL....1826
UVARM....1828
UWAVE and UEXTERNALDB....1831
VDISP....1834
VDLOAD....1835
VFILM....1837
VFRIC, VFRIC_COEF, and VFRICTION....1838
VHETVAL....1842
VSDVINI....1844
VUAMP....1845
VUCHARLENGTH....1847
VUCREEPNETWORK....1848
VUEL....1849
VUFIELD....1852
VUGENS....1853
VUHARD....1854
VUHYPER....1855
VUHYPER_STRETCH....1858
VUINTER....1859
VUINTERACTION....1862
VUMAT....1865
VUMATHT....1871
VUSDFLD....1872
VUTRSNETWORK....1873
VUVISCOSITY....1874
VWAVE....1875

Miscellaneous Modeling Options....1878

Activation functions used by feed-forward neural networks....1879
Elements tested....1879
Features tested....1879
Problem description....1879
Results and discussion....1879

Input files....1879
Figures....1880
Adaptive mesh for solid elements in Abaqus/Standard....1881
Elements tested....1881
Features tested....1881
Problem description....1881
Results and discussion....1881
Input files....1881
Adjusting nodal coordinates....1884
Elements tested....1884
Features tested....1884
Problem description....1884
Results and discussion....1884
Input files....1884
Amplitude....1885
Elements tested....1885
Features tested....1885
Problem description....1885
Results and discussion....1886
Input files....1887
Figures....1888
Spatially varying element properties....1896
Using distributions to define thicknesses, offsets, material orientations, and stiffness matrices for shell and membrane elements. 1896
Using distributions to define thicknesses, material orientations, and material behaviors for continuum elements. 1904
Using distributions while transferring results from one Abaqus/Standard analysis to another....1908
Using distributions for specifying orientation for connector elements....1911
Define a part instance that will be used for display only....1913
Problem description....1913
Results and discussion....1913
Input files....1913
Establishing geostatic equilibrium when the initial stress state is unknown....1914
Elements tested....1914
Features tested....1914
Problem description....1914
Results and discussion....1914
Input files....1914
Specifying geometric imperfection and parameter shape variation....1916
Elements tested....1916
Features tested....1916
Problem description....1916
Results and discussion....1916
Input files....1916
Specifying user-defined field variables directly in the input file....1918
Features tested....1918
Problem description....1918
Results and discussion....1918
Input files....1918
Inertia-based load balancing....1920

Automobile suspension component....1920
Assembly loading and free body motion of a rocket under thrust....1920
Stability analysis of a submerged structure (submarine)....1921
Inertia relief with multiple load cases....1922
Inertia relief of a model with substructures....1923
Defining the cutting surface....1924
Elements tested....1924
Features tested....1924
Problem description....1924
Results and discussion....1924
Input files....1924
Defining a matrix for part of a model....1926
Linear perturbation analysis of a truss model....1926
Multiple load case analysis of a beam model with an equation constraint and a multi-point constraint.1926
Large-sliding contact with node-based contact surface....1927
Three-dimensional model with predefined temperatures and distributed surface loads....1928
Tip loading of a diving board....1928
Natural frequency extraction of a diving board....1929
User-defined coordinate system....1930
Elements tested....1930
Features tested....1930
Problem description....1930
Remarks....1930
Results and discussion....1931
Input files....1932
Pre-tension section....1934
Elements tested....1934
Features tested....1934
Problem description....1934
Results and discussion....1934
Input files....1934
Radiation view factor definition: symmetries and blocking....1936
Infinitely long square section tube....1936
Infinitely long square section tube with blocking....1938
Finite length square section tube....1941
Square section tubular ring....1943
Infinitely extending three-dimensional array of cubic objects....1945
Infinitely long finned tube inside another infinitely long tube....1947
Release rotational degrees of freedom....1951
Elements tested....1951
Features tested....1951
Problem description....1951
Results and discussion....1951
Input files....1952
Extrapolation....1953
Elements tested....1953
Features tested....1953
Problem description....1953

Results and discussion....1953
Input files....1953
Surface-based fluid cavities....1954
Fluid behavior....1954
Gas structure interaction....1955
Fluid exchange....1955
Fluid inflators....1956
Defining temperature, field variable, pressure stress, and pore fluid pressure values....1957
Reading temperature and field variable data from results files....1957
Composite shell temperature loads....1962
Temperature interpolation to midside nodes....1963
Temperature interpolation between dissimilar meshes....1964
Transferring temperatures between dissimilar meshes with user-specified regions....1965
Reading temperature and pressure data from results files....1967
Reading solution-dependent variables from results files....1968
Reading scalar nodal output from the output database into field variables....1969
Specifying a known predefined pore fluid pressure field....1972
Coupled pore-thermal elements....1974
Elements tested....1974
Features tested....1974
Problem description....1974
Results and discussion....1974
Input files....1974
Product Index....1975
Abaqus/Standard....1976
Abaqus/Explicit....1987
Abaqus/CAE....1993
Abaqus/AMS....1994
Abaqus/Aqua....1995
Abaqus/Design....1996

Trademarks and Legal Notices

Trademarks

Abaqus, **3D**EXPERIENCE®, the 3DS logo, the Compass icon, IFWE, 3DEXCITE, 3DVIA, BIOVIA, CATIA, CENTRIC PLM, DELMIA, ENOVIA, GEOVIA, MEDIDATA, NETVIBES, OUTSCALE, SIMULIA and SOLIDWORKS are commercial trademarks or registered trademarks of Dassault Systèmes, a European company (Societas Europaea) incorporated under French law, and registered with the Versailles trade and companies registry under number 322 306 440, or its subsidiaries in the United States and/or other countries. All other trademarks are owned by their respective owners. Use of any Dassault Systèmes or its subsidiaries trademarks is subject to their express written approval.

Legal Notices

Abaqus and this documentation may be used or reproduced only in accordance with the terms of the software license agreement signed by the customer, or, absent such agreement, the then current software license agreement to which the documentation relates.

This documentation and the software described in this documentation are subject to change without prior notice.

Dassault Systèmes or its Affiliates shall not be responsible for the consequences of any errors or omissions that may appear in this documentation.

© Dassault Systèmes Americas Corp., 2026.

For a full list of the third-party software contained in this release, please go to the Legal Notices in the Abaqus 2026 HTML documentation, which can be obtained from a documentation installation, or in the SIMULIA Established Products 2026 Program Directory, which is available on *www.3ds.com*.

Abaqus Verification Guide

The Abaqus Verification Guide contains a large number of test cases that provide evidence that the implementation of the numerical model produces the expected results for one or several well-defined options in the code.
The test cases are sufficiently small that they can be executed efficiently while producing results that can be compared to independently calculated solutions or experimental results.

This guide is a part of the Abaqus® documentation collection, which describes all the capabilities of the Abaqus finite element analysis technology used in SIMULIA® applications.

In addition to this guide, there are two other guides that contain worked problems. The *Abaqus Benchmarks Guide* contains benchmark problems that provide evidence that the software can produce a result from a benchmark defined by an external body or institution such as NAFEMS. The tests in this guide are sufficient to show accuracy and convergence compared to benchmark data. The *Abaqus Example Problems Guide* contains many solved examples from which users can learn how to run simulations involving nontrivial physics.

The qualification process for new Abaqus releases includes running and verifying results for all problems in this guide, the *Abaqus Example Problems Guide*, and the *Abaqus Benchmarks Guide*.

It is important that a user become familiar with this guide, the *Abaqus Example Problems Guide*, and the *Abaqus Benchmarks Guide* before any analysis is done to determine the level of verification that has been done of the capabilities that will be used. The user should then decide whether any additional verification is necessary before starting the analysis.

Input files referred to in the guides are included with the Abaqus release in compressed archive files. To reproduce the graphical representation of the solution reported in some of the problems, modifications to various options (such as step time, display options, output requests, solver precision, output frequency, etc.) may be required.

The **abaqus fetch** utility is used to extract these input files for use. For example, to fetch input file *ec12afe1.inp* for *Eigenvalue extraction for single unconstrained elements*, type

```
abaqus fetch job=ec12afe1.inp
```

Parametric study script (`.psf`) and user subroutine (`.f`) files can be fetched in the same manner. All files for a particular problem can be obtained by leaving off the file extension. The **abaqus fetch** utility is explained in detail in *Fetching Sample Input Files*.

It is sometimes useful to search the input files. The **findkeyword** utility is used to locate input files that contain user-specified input. This utility is defined in *Querying the Keyword/Problem Database*.

What's New

This page describes recent changes in Abaqus Verification.

2026 FD04

Contact Surface Representation of Threads in Abaqus/Standard

New tests verify the internally generated surface-element representation of threads.

Benefits: The tests verify the capability in Abaqus/Standard.

The new capability improves contact simulation accuracy involving relatively large twisting motion between the external and the internal threads.

For more information, see *Contact surface representation of threads*.

Abaqus/Explicit: Cohesive Contact Involving Large Cohesive Strains

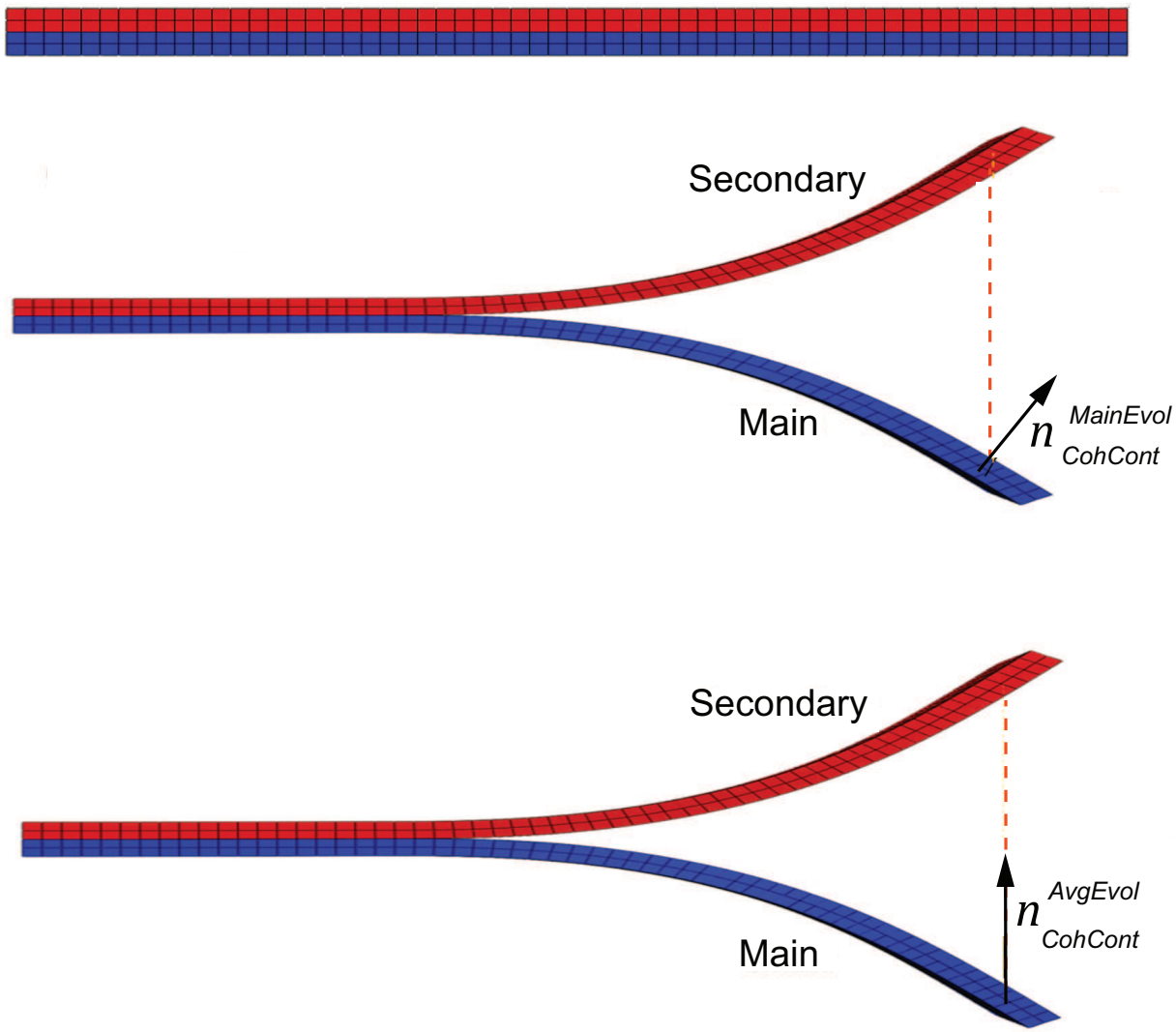
New tests verify cohesive contact behavior involving highly ductile cohesive bonds.

Benefits: The tests verify the capability in Abaqus/Explicit.

The figure below shows the initial and deformed configurations for a peeling test simulation with ductile cohesive contact bonds. The two parts are initially flat (top image) with cohesive contact along the entire length of the interface. Symmetry exists in the mesh refinement and loading conditions.

In the case shown in the middle image, the cohesive contact normal direction evolves with the main surface only, such that the cohesive strain develops normal and shear components. The dotted line in the image connects interacting points of a cohesive constraint.

In the case shown in the bottom image, the cohesive strain is aligned with the cohesive contact normal direction, which evolves taking changes in both the main and secondary surface normal directions into consideration. There is zero cohesive shear strain in this case.



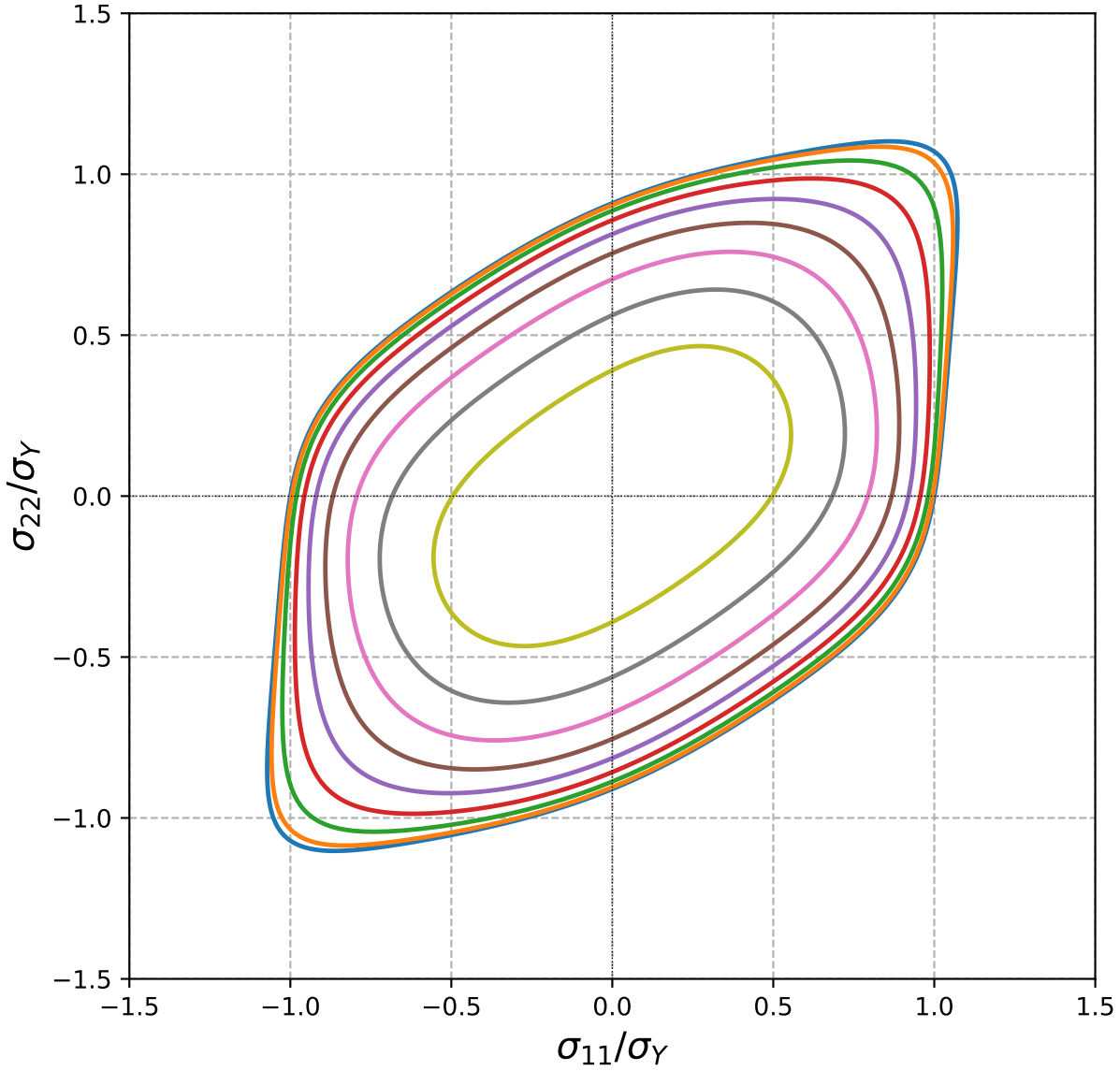
For more information, see *Cohesive surface interaction*.

Barlat Anisotropic Plasticity

New tests verify the Barlat plasticity models.

Benefits: The tests verify the capability in Abaqus/Standard and Abaqus/Explicit.

New tests are added to verify the modeling of Barlat Yld2004-18p and Barlat Yld91 yield surfaces using plane stress elements. Tests are also added to verify the new Barlat Yld2000-2d model. The plot below shows the yield surface for the 2090-T3 material predicted with the Barlat Yld2000-2d model.



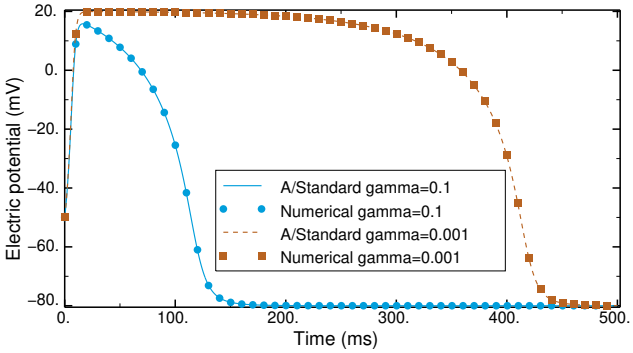
For more information, see *Barlat anisotropic plasticity*.

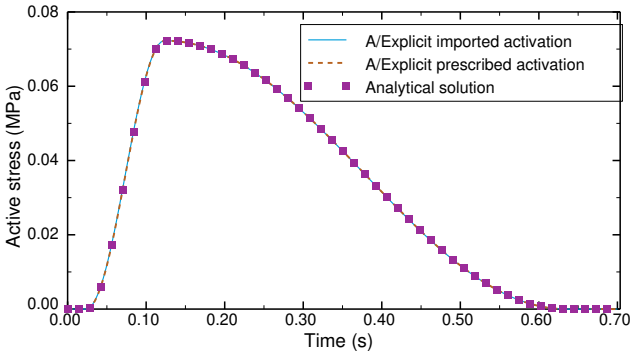
Electrical and Mechanical Modeling of Cardiac Tissue

New tests verify the electrical modeling of cardiac tissue in Abaqus/Standard and its mechanical modeling in Abaqus/Explicit.

Benefits: The tests verify the capabilities in Abaqus/Standard and Abaqus/Explicit.

The figures below show the agreement of the Abaqus/Standard and Abaqus/Explicit results with available numerical or analytical solutions.





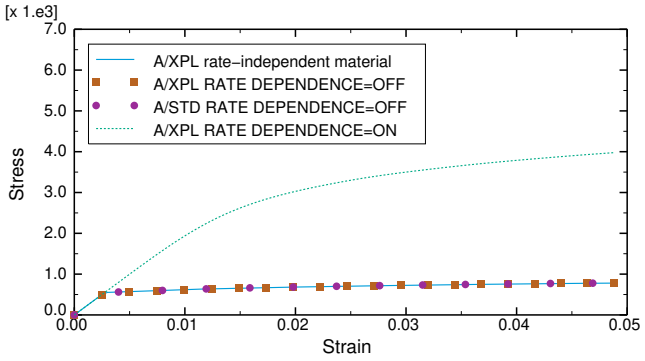
For more information, see *Electrical and mechanical modeling of cardiac tissue*.

Control of Material Rate-Dependence

New tests verify the control of material rate-dependence when running simulations in Abaqus/Standard and Abaqus/Explicit.

Benefits: The tests verify the capability in Abaqus/Standard and Abaqus/Explicit.

The figure below shows the agreement of the simulations ignoring material rate-dependent behaviors with reference simulations where the corresponding material definitions do not contain rate-dependent properties.



For more information, see *Controlling material rate dependence*.

Surface Tension Loads with Eulerian Elements

New tests demonstrate the application of surface tension loads with Eulerian elements.

Benefits: The tests verify the capabilities in Abaqus/Explicit.

For more information, see *Surface tension loads*.

Multiple Linear Load Cases in Submodel Analysis

A new test verifies that you can use multiple linear load cases in a submodel analysis.

Benefits: The test verifies the capability in Abaqus/Standard and Abaqus/Explicit.

For more information, see *Using Multiple Linear Load Cases in a Submodel* and *General submodeling*.

Time Synchronization in General Submodeling

New tests verify the default time synchronization between the global model and submodel.

Benefits: The tests verify the capability in Abaqus/Standard and Abaqus/Explicit.

For more information, see *Referring to the Step and Increment of the Global Model* and *General submodeling*.

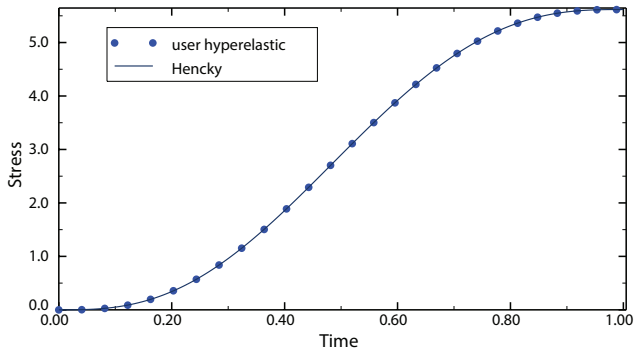
2026 FD03

`VUHYPER_STRETCH`

A new test is added to verify the user-defined stretch-based hyperelastic material model in Abaqus/Explicit.

Benefits: The test verifies the capabilities of `VUHYPER_STRETCH` in Abaqus/Explicit.

Stress results obtained from `VUHYPER_STRETCH` are validated against the solution using the Hencky hyperelastic material.



For more information, see *VUHYPER_STRETCH*.

Trajectory-Based Eigenstrain Technique for Additive Manufacturing Simulations

New verification tests demonstrate the application of eigenstrains during element activation in additive manufacturing–related simulations using the trajectory-based eigenstrain technique.

Benefits: The new tests verify the capabilities in Abaqus/Standard.

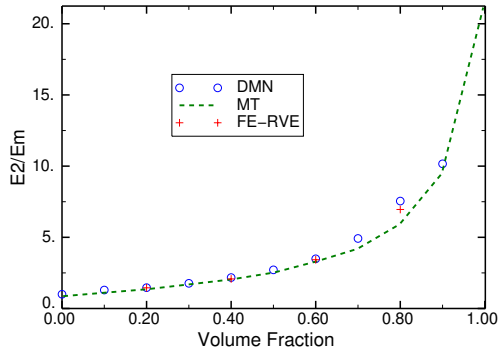
For more information, see *Trajectory-based eigenstrain analysis*.

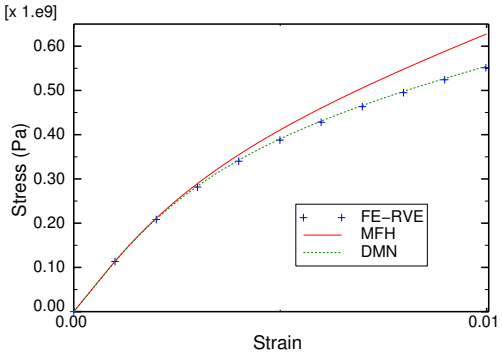
Multiscale Materials

New tests are added to verify the deep material networks (DMNs) for multiscale material modeling.

Benefits: The new tests verify the capabilities in Abaqus/Standard.

The new tests validate the DMN predictions against both FE-RVE simulations and mean-field homogenization models, spanning both linear and nonlinear composite behaviors.





For more information, see *Multiscale linear elastic materials*.

2026 FD02

ABD Stiffness Sensitivities in Adjoint Sensitivity Analyses

A new test verifies the ABD stiffness sensitivity calculations.

Benefits: The test verifies the capability in Abaqus/Standard.
The ABD stiffness sensitivities computed by Abaqus/Standard are compared to corresponding results obtained using the finite difference method.

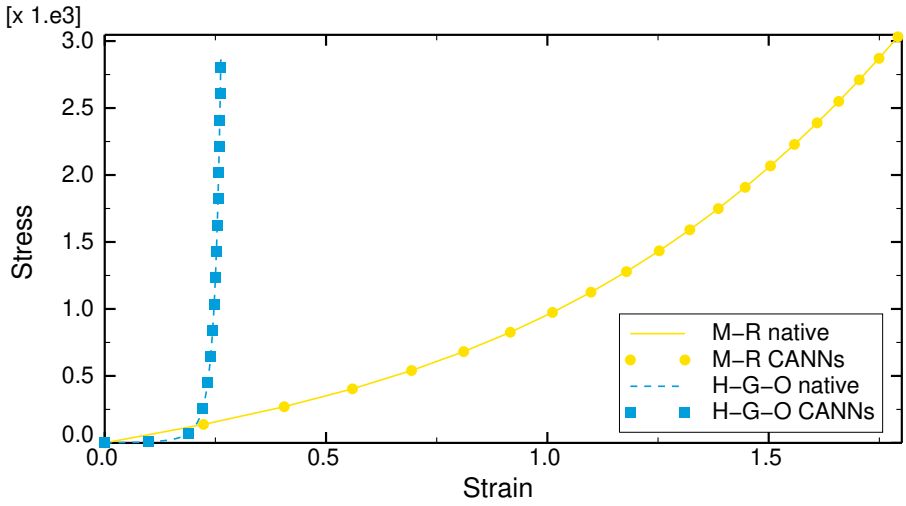
For more information, see *Adjoint Design Sensitivity Analysis Overview*, *Specifying Design Variables*, and *Adjoint design sensitivity analysis for static procedures*.

Constitutive Artificial Neural Networks (CANNs) Models

New tests verify the capability to define anisotropic hyperelastic models using constitutive artificial neural networks (CANNs) models.

Benefits: The tests verify the capability in Abaqus/Standard and Abaqus/Explicit.

The figure below shows the stress-strain response of a Mooney-Rivlin material and a Holzapfel-Gasser-Ogden material. The results for these models implemented as CANNs models agree with the results for their respective native implementations.



For more information, see *Constitutive Artificial Neural Networks (CANNs) models*.

Feed-Forward Neural Networks Evaluation

A new test verifies the capability of defining and evaluating multiple feed-forward neural networks.

Benefits: The test verifies the capability in Abaqus/Standard.
The new test demonstrates how to define and evaluate feed-forward neural networks from Abaqus/Standard user subroutines.

For more information, see *Activation functions used by feed-forward neural networks*.

Discrete Particle Swelling

A new test that verifies the evolution of the particle radius against a first-order model that is fitted to experimental data.

Benefits: The test verifies the capability in Abaqus/Explicit.

For more information, see *Discrete particle swelling*.

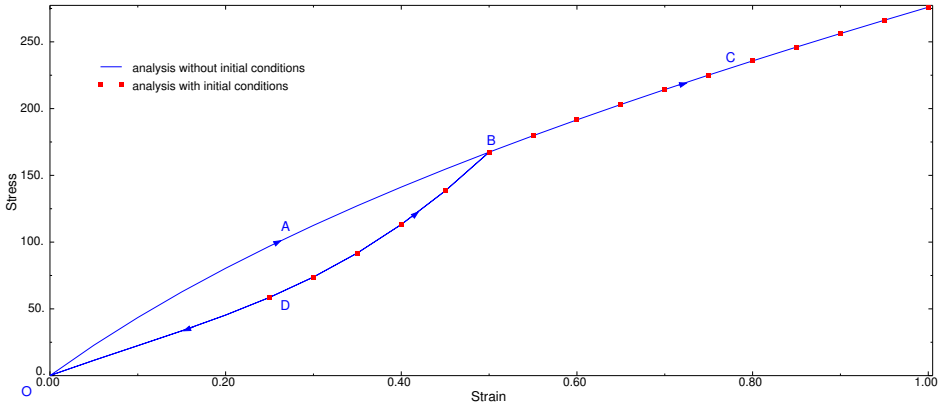
2026 FD01

Initial Stresses with Mullins Effect

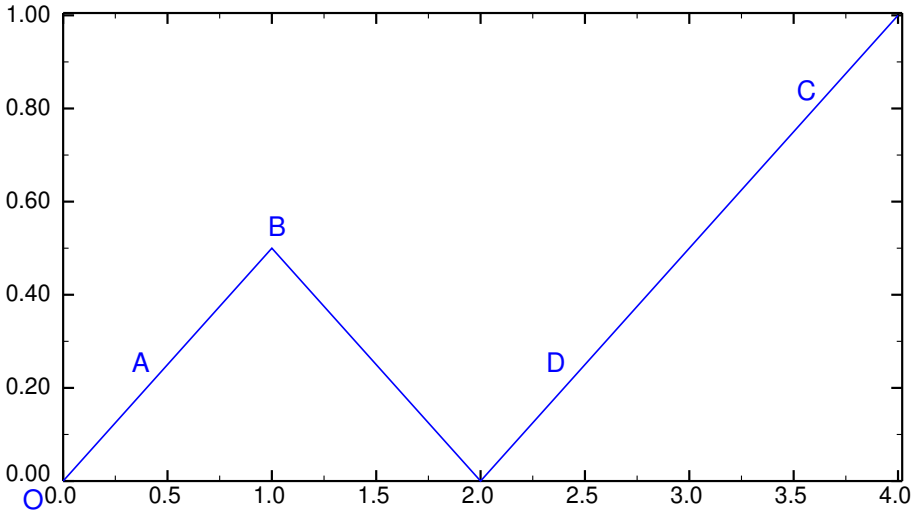
New tests verify the capability of prescribing initial stresses for materials with the Mullins effect model.

Benefits: The tests verify the capability in Abaqus/Standard and Abaqus/Explicit.

The first figure below compares stress-strain responses of two analyses in which specimens are subjected to nonmonotonic uniaxial loading.



The specimens are modeled using a hyperelastic material with the Ogden-Roxburgh Mullins effect model. In the first analysis, the material is subject to a nonmonotonic loading path as shown in the second figure.



In the second analysis, initial values of stress and maximum energy density are specified using results at point D from the first analysis as shown in the figures, and the load amplitude is the same as in the first analysis. The results of both analyses compare well.

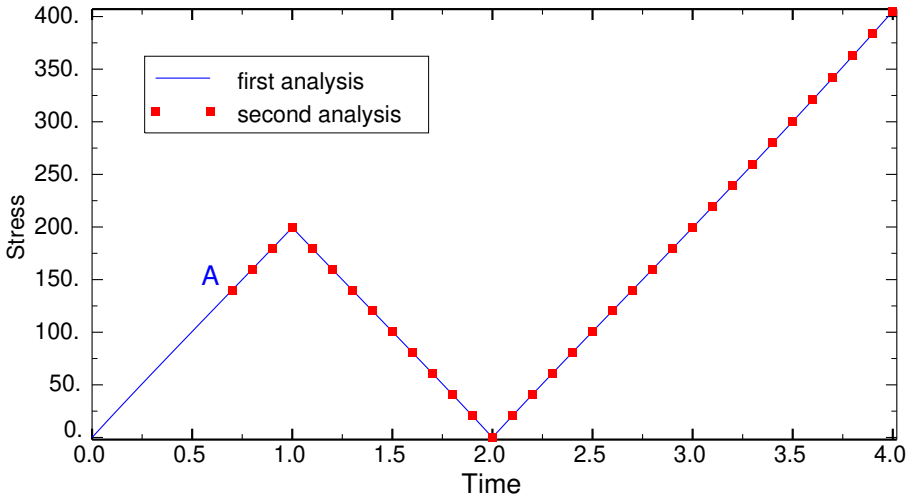
For more information, see *Defining Initial Stresses*.

Long-Term Initial Stresses

New tests verify the capability of prescribing long-term initial stresses.

Benefits: The tests verify the capability in Abaqus/Standard and Abaqus/Explicit.

The figure below compares stress-time histories of two analyses performed using viscoelastic materials.



In both cases small relaxation times are specified for each term in the Prony series that causes the stress to relax immediately. After the first analysis is completed, the second analysis uses the long-term initial stress capability to prescribe initial stresses using results from point A of the first analysis, and then the analysis continues with the same load applied. Both analyses produce identical results, which is expected in this case.

For more information, see *Defining Initial Stresses*.

C/C++ Interfaces to User Utility Routines

A new test with a user subroutine `UVAREL` written in C++ verifies C/C++ interfaces to utility subroutines `getElMPtVarC`, `getSecVarC`, and `getVrmC`.

Benefits: The tests verify the capability in Abaqus/Standard.

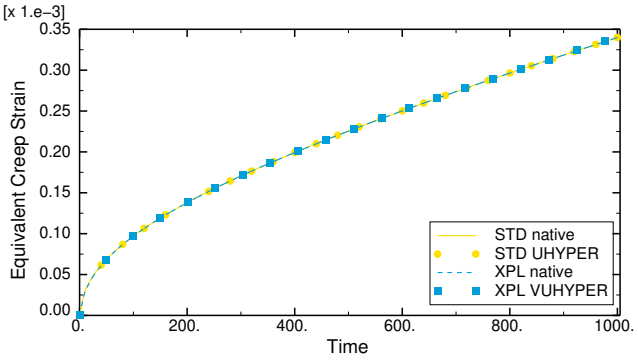
For more information, see *UVAREL*.

Viscous Softening

New tests verify the viscous softening capability in the parallel rheological framework.

Benefits: The tests verify the capability in Abaqus/Standard and Abaqus/Explicit.

The figure below shows the stress-strain response of a nonlinear viscoelastic material subjected to uniaxial loading and unloading and undergoing viscous softening.



Results given by Abaqus/Standard and Abaqus/Explicit are in good agreement with each other and with an equivalent implementation using the `UHYPER` and `VUHYPER` user subroutines.

For more information, see *Nonlinear large-strain viscoelasticity with hyperelasticity*.

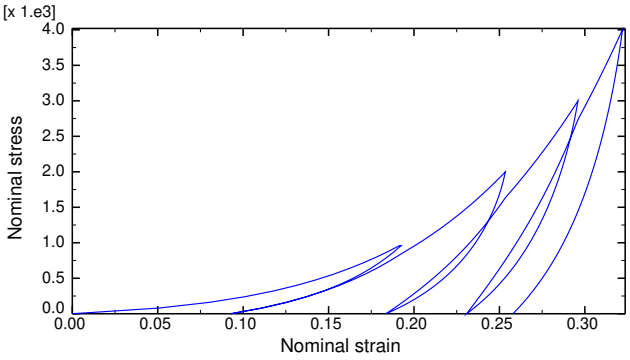
2026 GA

Mullins Effect with Hyperfoam Material and Large-Strain Viscoelasticity

New tests verify the Mullins effect modeling capabilities with large-strain time-domain viscoelasticity and a hyperfoam material.

Benefits: The tests verify the capability in Abaqus/Standard.

The figure below shows the stress-strain response of a specimen subjected to cycling uniaxial loading. The specimen is modeled using the hyperfoam material with large-strain viscoelasticity and the Ogden-Roxburgh Mullins effect model. The response shows that the material can capture the hysteresis as well as the stress-softening on unloading, which are exhibited by many polymers subjected to cyclic loading.



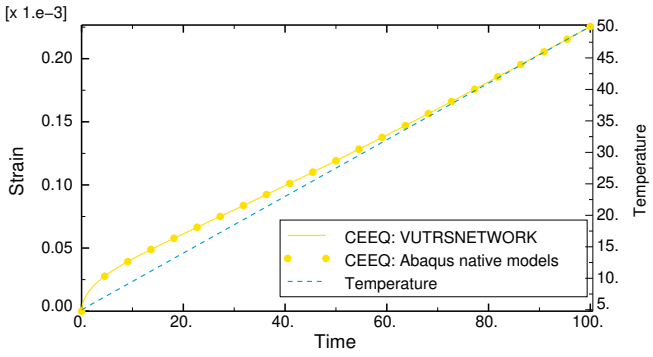
For more information, see *Stress Softening in Rubberlike Materials*.

`VUTRSNETWORK`

New tests verify the variables that are passed into user subroutine VUTRSNETWORK and compare results with those of equivalent temperature-time shifts available in Abaqus/Explicit.

Benefits: The tests verify the capability in Abaqus/Explicit.

The figure below shows the time history of creep strain and prescribed temperature from the new verification test. The results obtained using the VUTRSNETWORK subroutine agree with those obtained using native Abaqus models.



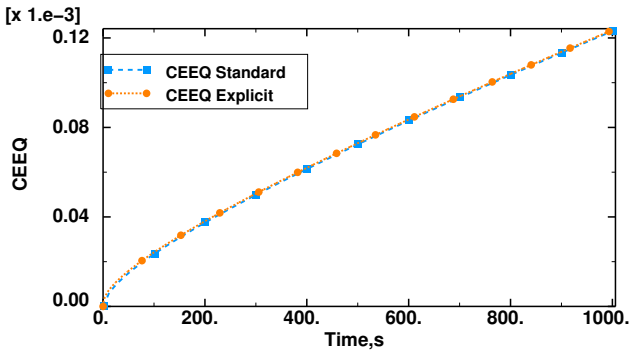
For more information, see *VUTRSNETWORK*.

Anand Model in a Parallel Rheological Framework

New tests verify the Anand creep model used in a parallel rheological framework.

Benefits: The tests verify the capability in Abaqus/Standard and Abaqus/Explicit.

The figure below shows the history output of creep strain from one of the new uniaxial creep tests. The Anand law is used to model the creep behavior of one of the four viscoelastic networks. Results given by Abaqus/Standard and Abaqus/Explicit are in good agreement.



For more information, see *Parallel Rheological Framework* and *Nonlinear large-strain viscoelasticity with hyperelasticity*.

2025 FD03

Contact Wear Modeling

New tests verify the generation of a 3D model using symmetric model generation.

Benefits: The tests verify the capability in Abaqus/Standard and Abaqus/Explicit.

For more information, see *Contact wear modeling*.

Kamal Cure Kinetics Model

New tests verify the cure modeling capabilities using the Kamal model with field-variable dependent parameters.

Benefits: The tests verify the capability in Abaqus/Standard.

For more information, see *Curing Processes in Polymers* and *Cure modeling in Abaqus/Standard*.

Ductile Damage Initiation Criteria

New tests verify the evaluation of ductile damage initiation criteria for hyperelastic materials with isotropic plasticity.

Benefits: The tests verify the capability in Abaqus/Standard.

For more information, see *Progressive Damage and Failure* and *Progressive damage and failure of ductile metals*.

Transferring Results with Nodal Temperature and Field Variables

New tests verify the transfer of results with nodal temperature and field variables from an Abaqus/Standard analysis to an Abaqus/Explicit analysis, from an Abaqus/Explicit analysis to an Abaqus/Standard analysis, and from an Abaqus/Explicit analysis to an Abaqus/Explicit analysis.

Benefits: The tests verify the capability in Abaqus/Standard and Abaqus/Explicit.

For more information, see *Transferring results with nodal temperature and field variables*.

Transferring Results of an Analysis Model Multiple Times

New tests verify the transfer of results of solid element sets multiple times from an Abaqus/Standard analysis to an Abaqus/Explicit analysis and from an Abaqus/Explicit analysis to an Abaqus/Explicit analysis.

Benefits: The tests verify the capability in Abaqus/Standard and Abaqus/Explicit.

For more information, see *Transferring results of an analysis model multiple times*.

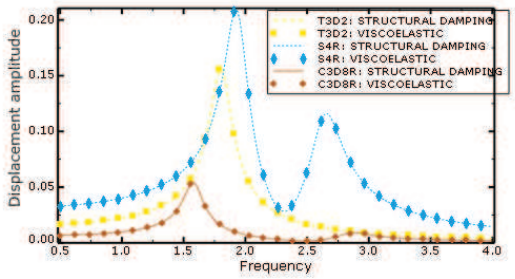
2025 FD02

Structural Damping

New tests verify constant and frequency-dependent structural damping.

Benefits: The tests verify the capability in Abaqus/Standard.

A steady-state dynamic analysis is conducted to determine the steady-state response of 1-element structures subjected to uniaxial harmonic excitation. The following figure shows results for a linear elastic material with structural damping and for a material with structurally equivalent viscoelasticity without structural damping. The results obtained using the two different approaches agree very well with each other.



For more information, see *Structural damping* and *Structural Damping*.

Adjoint Design Sensitivity Analysis for Combined Abaqus and fe-safe–Based Fatigue Procedures

A new test verifies the adjoint fatigue design responses and corresponding sensitivity analysis techniques for shell and solid continuum elements for a combined Abaqus and fe-safe–based fatigue procedure.

Benefits: The test verifies the capability in Abaqus/Standard.

For more information, see *Adjoint design sensitivity analysis for combined Abaqus and fe-safe–based fatigue procedures*.

Mullins Effect

New tests verify the Mullins effect modeling capabilities with the exponential and slow decay forms used to define the evolution of the damage variable.

Benefits: The tests verify the capability in Abaqus/Standard and Abaqus/Explicit.

For more information, see *Mullins effect and permanent set*.

2025 FD01

Paperboard Material Model

New tests verify the modeling of paperboard material.

Benefits: The tests verify the capability in Abaqus/Standard and Abaqus/Explicit.

For more information, see *Paperboard material model*.

UMAT and UHYPER

New tests verify that the correct value of the network identifier is passed into user subroutine UHYPER when the subroutine is used with the parallel rheological framework.

Benefits: The tests verify the capability in Abaqus/Standard

For more information, see *UMAT and UHYPER*.

UHYPER_STRETCH

New tests verify that the correct value of the network identifier is passed into user subroutine UHYPER_STRETCH when the subroutine is used with the parallel rheological framework.

Benefits: The tests verify the capability in Abaqus/Standard

For more information, see *UHYPER_STRETCH*.

VUHYPER

New tests verify the variables that are passed into user subroutine VUHYPER and compare results with those of equivalent hyperelastic material models available in Abaqus/Explicit.

Benefits: The tests verify the capability in Abaqus/Explicit.

For more information, see *VUHYPER*.

Normal Definitions of Beams and Shells

New tests verify the use of B21, B31, B32OSH, and B33H elements with user-specified nodal normals.

Benefits: The tests verify the capability in Abaqus/Standard and Abaqus/Explicit.

For more information, see *Normal definitions of beams and shells*.

Elastic Materials

New tests verify the hyperelasticity with Hencky strain energy function in Abaqus/Explicit.

Benefits: The tests verify the capability in Abaqus/Explicit.

For more information, see *Elastic materials*.

Miscellaneous Results Transfer Tests

New tests verify the results of transfer tests.

Benefits: The tests verify the capability in Abaqus/Standard and Abaqus/Explicit.
Tests verify the transfer of the state at the fluid cavity reference node when the material state is imported and the transfer of the state of contact pairs for models that uses part instance.

For more information, see *Miscellaneous results transfer tests*.

Coupling Constraints

New tests verify that appropriate constraints are formed for distributing couplings with collinear cloud nodes with rotational degrees of freedom that are allowed to participate in the coupling.

Benefits: The tests verify the capability in Abaqus/Standard and Abaqus/Explicit.

For more information, see *Coupling constraints*.

Contact Wear Modeling

New tests verify the contact wear capability for general contact.

Benefits: The tests verify the capability in Abaqus/Standard and Abaqus/Explicit.

For more information, see *Contact wear modeling*.

PDF Guides Available

You can download PDF versions of the Abaqus guides from each guide overview.

Benefits: You can easily access the PDF versions of the guides.
In earlier releases, PDF versions of the guides were updated only for the GA (General Availability) release and could be downloaded from the Dassault Systèmes Knowledge Base. As of 2025 FD01, the PDF guides will be updated with each release of the HTML versions of the guides.

For more information, see *Abaqus Verification Guide*.

2025 GA

Contact Wear Modeling

New tests verify step cycling controls for general contact.

Benefits: The tests verify the capability in Abaqus/Explicit.

For more information, see *Contact wear modeling*.

2024 FD04

Contact Wear Modeling

New tests verify the contact wear capability for general contact.

Benefits: The tests verify the capability in Abaqus/Explicit.

For more information, see *Contact wear modeling*.

2024 FD03

UVAREL

New tests verify variables that are passed into user subroutine `UVAREL` and integration point quantities that are read by utility subroutine `getElMPtVar`.

Benefits: The tests verify the capability in Abaqus/Standard.

For more information, see *UVAREL*.

Piezoresistive Effect

New tests model one-dimensional electrical current flow using axisymmetric and three-dimensional elements.

Benefits: The tests verify the capability in Abaqus/Standard.

For more information, see *Piezoresistivity properties: Piezoresistive effect*.

Bending Stabilization

New tests verify the bending stabilization capability.

Benefits: The tests verify the capability in Abaqus/Standard.

For more information, see *Bending stabilization for full-integration linear elements* and *Using Section Controls for Bending Stabilization*.

User-Defined Element Output Variables

New tests verify the definition and output of user-defined element variables.

Benefits: The tests verify the capability in Abaqus/Standard.

For more information, see *UVAREL* and **ELEMENT USER OUTPUT VARIABLES*.

Thermal Expansion

New tests verify the tangent and secant thermal expansion capability.

Benefits: The tests verify the capability in Abaqus/Standard.

For more information, see *Thermal Expansion*.

Stretching of Solid Body Using SPH Element Conversion

New tests verify the accuracy of Lagrangian element (C3D8R) to SPH particle (PC3D) conversion for a solid body subjected to uniaxial stretching.

Benefits: The tests verify the capability in Abaqus/Explicit.

For more information, see *Smoothed particle hydrodynamic analysis*.

Progressive Damage and Failure in Fiber-Reinforced Materials

A new Python test models progressive damage and failure in fiber-reinforced materials with multiscale modeling.

Benefits: The test verifies the capability in Abaqus/Standard.

For more information, see *Progressive damage and failure in fiber-reinforced materials*.

2024 FD02

Undrained Pore Fluid Flow and Stress Analysis

New tests verify undrained pore fluid flow and stress analysis.

Benefits: The tests verify the capability in Abaqus/Explicit.
A unit cube of geometry with distorted meshes is used to test three-dimensional solid elements, and a simple rectangular geometry is used to test plane strain elements as well as axisymmetric solid elements. The medium is subject to prescribed displacement boundary conditions, and the numerical results are compared with a reference solution.

For more information, see *Undrained pore fluid flow and stress analysis*.

2024 FD01

Progressive Damage and Failure in Fiber-Reinforced Materials

New tests contain basic cases to model progressive damage and failure in fiber-reinforced materials with multiscale modeling.

Benefits: The tests verify the capability in Abaqus/Standard.

For more information, see *Progressive damage and failure in fiber-reinforced materials*.

Generalized Damage and Failure for Anisotropic Materials

New tests contain basic cases with various damage initiation criteria for anisotropic materials.

Benefits: The tests verify the capability in Abaqus/Standard and Abaqus/Explicit.

For more information, see *Generalized damage and failure for anisotropic materials*.

Field Expansion

New tests verify the field expansion capability for both solid and pore fluid phases in a porous medium.

Benefits: The tests verify the capability in Abaqus/Standard.
The solid grains and the pore fluid in a porous medium are subjected to thermal expansion and multiple field expansions. Both built-in capabilities and user subroutine *UEXPAN* are used.

For more information, see *Field expansion test*, *UEXPAN*, and *UEXPAN*.

Band-Limited Damping in Abaqus/Standard

New tests verify the use of band-limited damping with various element types and material models.

Benefits: The tests verify the capability in Abaqus/Standard.

For more information, see *Material Damping*.

Updated SLIPRING Contact Angle in Abaqus/Standard

New tests verify the use of the updated contact angle (based on the nodal coordinates) for the SLIPRING connection type in Abaqus/Standard.

Benefits: The tests verify the capability in Abaqus/Standard.

For more information, see *Frictional SLIPRING connectors*.

Contact Wear Modeling

New tests verify the contact wear capability for both general contact and contact pairs.

Benefits: The tests verify the capability in Abaqus/Standard.

For more information, see *Contact wear modeling*.

2024 GA

Tie Constraints

New tests verify that the configuration used for the formation of tie constraints in an Abaqus/Explicit import analysis depends on whether or not there are changes to the definitions of the surfaces to be tied.

Benefits: The tests verify the capability in Abaqus/Explicit.

For more information, see *Miscellaneous results transfer tests*.

Analysis Procedures and Techniques Verification

This section defines the basic tests used to provide evidence that the implementation of some of the procedure options in the Abaqus procedures library produces the expected results. Some of these tests also verify the capability to extract output from the restart (`.res`) file and to write new results to output files using a postprocessing analysis procedure—see the problem descriptions for details.

In this section:

- *Dynamic analysis*
- *Crack propagation*
- *Substructuring*
- *Additive manufacturing process simulation*
- *Electromagnetic analysis*
- *Coupled thermal-electrochemical analysis*
- *Piezoelectric analysis*
- *Submodeling*
- *Acoustic and shock analyses*
- *Model change*
- *Symmetric model generation and analysis of cyclic symmetry models*
- *Abaqus/Aqua analysis*
- *Design sensitivity analysis*
- *Transferring results between Abaqus/Standard and Abaqus/Explicit*
- *Meshed beam cross-sections*
- *Co-simulation*
- *Adaptive remeshing*
- *Discrete particle methods*
- *Miscellaneous procedures and techniques*

Dynamic analysis

In this section:

- *Modal dynamic analysis with baseline correction*
- *Steady-state dynamic analysis for two-dimensional elements*
- *Steady-state dynamic analysis for infinite elements*
- *Random response analysis*
- *Single degree of freedom spring-mass systems*
- *Linear kinematics element tests*
- *Mass scaling*

Modal dynamic analysis with baseline correction

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

B23 CAX4H

Features tested

Baseline correction of simple accelerogram record.

Primary and secondary base motions.

Problem description

The first example (*pmodbase.inp*, *pmodbas2.inp*, and *pmodbas2a.inp*) is a modal dynamic, time history analysis that is performed on a one-element cantilever structure using a B23 element. As the base motion record, a simple sine-shaped accelerogram is assumed for the time of one sine period. The record is corrected for the total time of the record duration. The choice of the base motion record as a sine function allows the analytical calculation of the parabolic correction to the record using the formulæ from *Baseline correction of accelerograms*. The values of the three constants for the parabolic correction are c_1 = –0.8308, c_2 = 0.4207, and c_3 = 2.1717; and the corrected accelerogram is

$$a_c(t) = \sin(t) - 0.8308 + 0.4207\frac{t}{2\pi} + 2.1717\frac{t}{2\pi},$$

where $0 < t < 2\pi$. Integrating twice gives the corresponding displacement record:

$$u_c(t) = t - \sin(t) - 0.8308\frac{t^2}{2} + 0.4207\frac{t^3}{12\pi} + 2.1717\frac{t^4}{48\pi^2}.$$

The second example (*pmodbas3.inp* and *pmodbas4.inp*) illustrates the application of multiple base motions in a time history modal dynamic analysis in which part of the structure is fixed while another part of it is subjected to excitation. The structure analyzed is a quarter-symmetry axisymmetric model of a cylinder made of rubberlike material. An 8 × 8 mesh with CAX4H elements is employed for the analysis. The structure is first preloaded in compression statically in the axial direction by a rigid platen, which is modeled as a rigid surface in *pmodbas3.inp* and as a rigid body in *pmodbas4.inp*; perfect bonding between the platen and the top surface of the cylinder is assumed. The response to applied axial (acceleration) excitation at the rigid surface reference node is sought. The acceleration records are the same as those used in the first problem. Since both fixed boundary conditions and applied acceleration boundary conditions occur in the same global (axial) direction in different parts of the structure, we treat the fixed boundary conditions as a primary base motion and the applied accelerations as a secondary base motion.

Results and discussion

The results for the first example are confirmed by running the input files *pmodbase.inp*, *pmodbas2.inp*, and *pmodbas2a.inp* and postprocessing the results file output. Although the three models differ in their “base” organizations—namely, the base in the first input file is handled as a *primary* base and that in the second and third input files is handled as a *secondary* base—the results they generate are identical. The plot of the total

displacement of the cantilever tip will show the considerable difference between the uncorrected and corrected records.

The results obtained for the second example by the two different input files, *pmodbas3.inp* and *pmodbas4.inp*, are the same.

Input files

pmodbase.inp	Invokes **BASE MOTION* without the BASE NAME parameter.
pmodbas2.inp	Invokes **BASE MOTION* with the BASE NAME parameter.
pmodbas2a.inp	Invokes **BASE MOTION* with TYPE=DISPLACEMENT.
pmodbas3.inp	Tests multiple base motions and rigid surface.
pmodbas4.inp	Tests multiple base motions and rigid element.

Steady-state dynamic analysis for two-dimensional elements

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

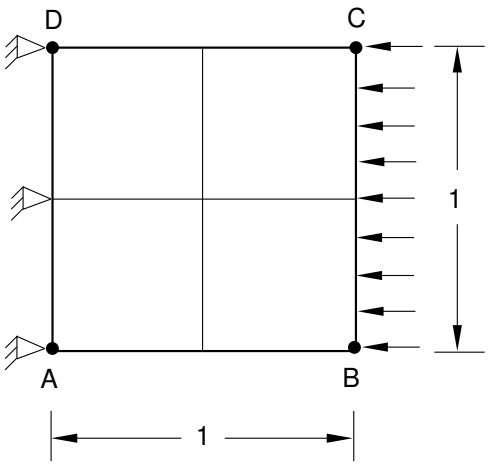
Elements tested

CPE3 CPE3H CPE4 CPE4H CPE4I CPE4IH CPE4R CPE4RH CPE6 CPE6H CPE6M CPE6MH CPE8 CPE8H CPE8R CPE8RH CPS3 CPS4 CPS4I CPS4R CPS6 CPS6M CPS8 CPS8R

Features tested

Direct-solution and subspace-based steady-state dynamic analysis of two-dimensional elements with damping.

Problem description



The model consists of a square structure that is fixed at edge AD and has a forced harmonic pressure applied at edge BC. Material damping is provided in the form of mass and stiffness proportional damping.

Material: Young's modulus = 20 GPa, Poisson's ratio = 0, density = 8000 kg/m^3.

Boundary conditions: $u_x = u_y = 0$ at end AD.

Damping: $\alpha = 5.36$, $\beta = 7.46 \times 10^{-5}$.

Forcing function: (steady-state harmonic)

$F = F_0 \sin(\omega t)$

F_0 = 30,000 N/m on edge BC

$\omega = 2\pi f$ Hz

f = 10 to 15 Hz

Reference solution

The results are confirmed by comparing them to a mode-based steady-state dynamic analysis using CPS4 elements.

Results and discussion

	Peak displacement	Peak stress	Frequency
	(mm)	(N/mm^2)	(Hz)
Reference solution	16.90	0.478	12.18
3-node elements	17.55	0.481	12.07
6-node elements	16.46	0.539	12.47
6-node modified elements	16.85	0.536	12.37
4-node elements	16.92	0.478	12.17
8-node elements	16.45	0.540	12.47

Input files

pssdce3sf.inp	CPE3 elements.
pssdce3sh.inp	CPE3H elements.
pssdce4sf.inp	CPE4 elements.
pssdce4sh.inp	CPE4H elements.
pssdce4si.inp	CPE4I elements.
pssdce4sj.inp	CPE4IH elements.
pssdce4sr.inp	CPE4R elements.
pssdce4sy.inp	CPE4RH elements.
pssdce6sf.inp	CPE6 elements.
pssdce6sh.inp	CPE6H elements.
pssdce6sk.inp	CPE6M elements.
pssdce6sl.inp	CPE6MH elements.
pssdce8sf.inp	CPE8 elements.
pssdce8sh.inp	CPE8H elements.
pssdce8sr.inp	CPE8R elements.
pssdce8sy.inp	CPE8RH elements.
pssdcs3sf.inp	CPS3 elements.
pssdcs4sf.inp	CPS4 elements.
pssdcs4si.inp	CPS4I elements.
pssdcs4sr.inp	CPS4R elements.

pssdcs6sf.inp	CPS6 elements.
pssdcs6sk.inp	CPS6M elements.
pssdcs8sf.inp	CPS8 elements.
pssdcs8sr.inp	CPS8R elements.
pssdmcs4sf.inp	Reference mode-based steady-state dynamic analysis.

Steady-state dynamic analysis for infinite elements

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

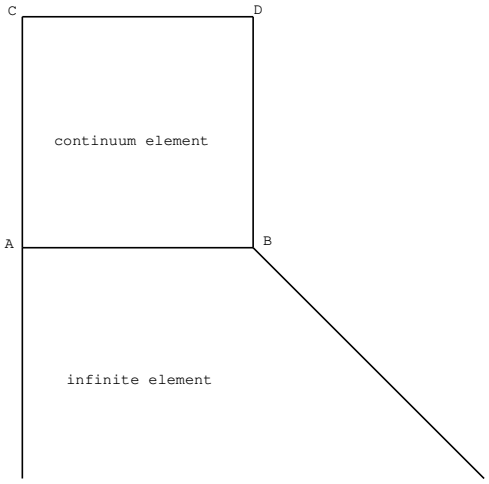
Elements tested

CIN3D8 CIN3D12R CIN3D18R CINAX4 CINAX5R CINPE4 CINPE5R CINPS4 CINPS5R

Features tested

Direct-solution steady-state dynamic analysis of infinite elements.

Problem description



The model consists of a single infinite element connected to one regular continuum finite element. The model is subjected to a plane wave and a shear wave. The results from this analysis are compared with a reference solution obtained from a model in which the infinite element is replaced by dashpots attached to the regular continuum element at points A and B. The damping coefficient corresponding to a plane wave, d_p, is computed as $d_p = \rho c_p$, where $c_p = \sqrt{\frac{\lambda + 2G}{\rho}}$ is the plane wave speed. Similarly, the damping coefficient corresponding to a shear wave, d_s, is computed as $d_s = \rho c_s$, where $c_s = \sqrt{\frac{G}{\rho}}$ is the shear wave speed.

Material: Young's modulus = 1.0, Poisson's ratio = 0.1, density = 0.01.

Boundary conditions: Plane wave: $u_y = 1.0 \times 10^{-4}$ along edge CD, $u_x = 0$ throughout the model.

Shear wave: $u_x = 1.0 \times 10^{-4}$ along edge CD, $u_y = 0$ throughout the model.

Results and discussion

The results are confirmed by comparing them to a direct-solution steady-state dynamic analysis of the model in which the infinite elements are replaced by dashpots. The displacements and phase angles match the reference solution in all cases.

Input files

ec38ifxw.inp	CIN3D8 elements.
ec38ifxt.inp	CIN3D8 elements replaced by dashpots.
ec3dirxw.inp	CIN3D12R elements.
ec3dirxt.inp	CIN3D12R elements replaced by dashpots.
ec3eirxw.inp	CIN3D18R elements.
ec3eirxt.inp	CIN3D18R elements replaced by dashpots.
eca4ifxw.inp	CINAX4 elements.
eca4ifxt.inp	CINAX4 elements replaced by dashpots.
eca5irxw.inp	CINAX5R elements.
eca5irxt.inp	CINAX5R elements replaced by dashpots.
ece4ifxw.inp	CINPE4 elements.
ece4ifxt.inp	CINPE4 elements replaced by dashpots.
ece5irxw.inp	CINPE5R elements.
ece5irxt.inp	CINPE5R elements replaced by dashpots.
ecs4ifxw.inp	CINPS4 elements.
ecs4ifxt.inp	CINPS4 elements replaced by dashpots.
ecs5irxw.inp	CINPS5R elements.
ecs5irxt.inp	CINPS5R elements replaced by dashpots.

Random response analysis

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

The tests in this section verify the random response capability for structures subjected to correlated and uncorrelated excitations. The tests include excitation from base motion and from concentrated and distributed loads.

Cantilever beam excited by base motion

Elements tested

B21

Features tested

Correlated and uncorrelated random base motions.

Problem description

A two-element cantilever beam aligned along the y-axis is excited by prescribed ground accelerations in global degrees of freedom 1 and 6. B21 elements of unit length are used. A white noise power spectral density is used to describe the applied ground accelerations.

Since random response analysis is a modal-based procedure, a frequency step is required to obtain the mode shapes and natural frequencies of the system. Steps 2 and 3 test correlated and uncorrelated excitation between global degrees of freedom 1 and 6, respectively. Steps 4 and 5 test arbitrary load case numbering. Only the first two mode shapes have been used in the random response analysis steps, with a damping ratio of 0.01 for each mode.

Results and discussion

For this problem the response power spectral densities can be relatively easily evaluated by hand calculations. The power spectral densities at various frequencies (including the natural frequencies) agree with the hand calculations. The results for Steps 2 and 4 should be identical to each other, as should the results for Steps 3 and 5.

Input files

prrbase.inp	Cantilever beam excited by base motion.

Cantilever beam excited by random concentrated and distributed loads

Elements tested

B21

Features tested

Correlated and uncorrelated random concentrated loads.

Multiple power spectral definitions, correlation definitions, and load cases.

Problem description

A two-element cantilever beam aligned along the x-axis is excited by transverse distributed and concentrated loads. The concentrated loads are applied at the free end (magnitude of -1.0) and at the midnode (magnitude of -2.0). The distributed load acts on the element closest to the cantilevered end (magnitude of 4.0). B21 elements of unit length are used. Both the distributed load and the concentrated loads are described by white noise power spectral densities.

Since random response analysis is a modal-based procedure, a frequency step is required to obtain the mode shapes and natural frequencies of the system. Steps 2 and 3 test correlated and uncorrelated concentrated loads, respectively. Steps 4 and 5 test arbitrary load case numbering. Only the first two mode shapes have been used in the random response steps, with a damping ratio of 0.01 for each mode.

Results and discussion

For this problem the response power spectral densities can be relatively easily evaluated by hand calculations. The power spectral densities at various frequencies (including the natural frequencies) agree with the hand calculations. The results for Steps 2 and 4 should be identical to each other, as should the results for Steps 3 and 5.

Input files

prrforc.inp	Cantilever beam excited by random concentrated and distributed loads.

Single degree of freedom spring-mass systems

Products: Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

CPE4R SPRINGA MASS DASHPOTA

Features tested

Time integration procedure, nonlinear springs and dashpot, distributed loads, point loads, gravity loading.

Problem description

There are six individual single degree of freedom spring-mass systems defined in this problem. In each case two springs are attached to a single CPE4R element that is constrained to have only vertical motion. The meshes are shown in *Figure 1*. The following cases are considered:

1. This single degree of freedom oscillator is loaded with a distributed load of 10^6 on the top of the element. The springs are linear, each with a stiffness of 2.0×10^6. The static displacement under this load is 0.25. The mass of the element is 1000. The analytical solution gives a period of 0.0993.
2. This single degree of freedom oscillator should be identical to Case 1. The springs are defined as nonlinear springs, but the tabular definition gives the same linear stiffness as the springs in Case 1. In this case the element is loaded with concentrated loads equal to the distributed load of Case 1.
3. The solution to this problem should be identical to that defined for Case 1. In this case the load is applied as a gravity load instead of as a distributed load. The springs are linear.
4. The definition of this problem is the same as that for Case 1 except that two point masses (mass of 500 each) are added to the problem. The addition of the point masses increases the period of this case to 0.1405.
5. In this single degree of freedom system the springs are nonlinear. Each spring has the same stiffness as the linear springs in Case 1 up to the static deflection of 0.5. Above a deflection of 0.5 the stiffness is 20 percent of the linear stiffness. The solution should be identical to Case 1 up to a displacement of 0.25. Because the nonlinear spring is not as stiff as the linear springs above a displacement of 0.25, the period of the oscillation in this case is greater than that of Case 1.
6. This single degree of freedom oscillator should be identical to Case 1 except for the added dashpot. The springs are defined as nonlinear springs, but the tabular definition gives the same linear stiffness as the springs in Case 1. In this case the element is loaded with concentrated loads equal to the distributed load of Case 1. A linear dashpot is attached parallel to the left spring.

Results and discussion

Figure 2 shows the displacement of each single degree of freedom system as a function of time. Cases 1, 2, and 3 have identical solutions and match the analytical solution for the single degree of freedom system. Case 6 shows smaller amplitudes of oscillation due to the damping effect of the dashpot. Case 4 matches the analytical solution for the added mass. Case 5 has no analytical solution; however, the results are qualitatively correct.

Input files

springs.inp	Input data used in this analysis.
springstfv.inp	A slightly modified version of input file springs.inp; this file includes temperature- and field-variable-dependent behavior for spring constants and dashpot coefficients.

Both input files are designed to provide identical results.

Figures

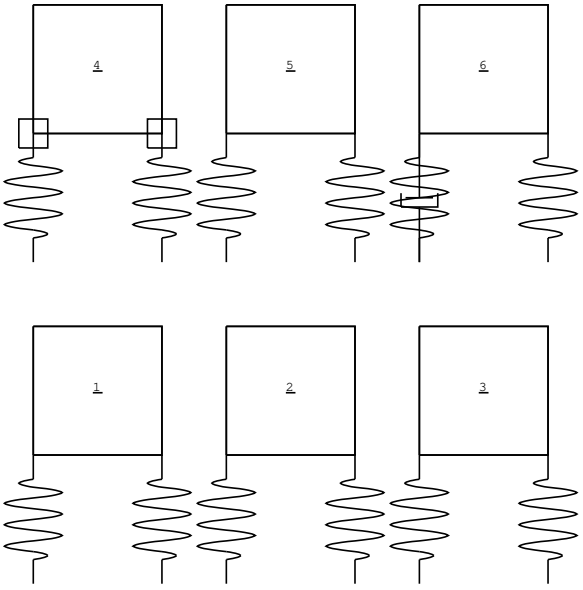


Figure 1: Single degree of freedom systems.

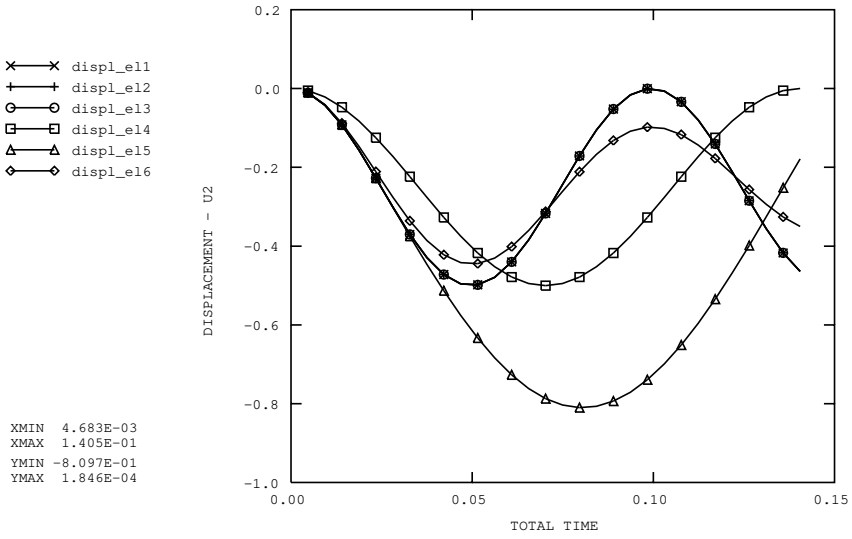


Figure 2: Displacements of single degree of freedom systems.

Linear kinematics element tests

Products: Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

B21 B22 B31 B32 C3D8 C3D8I C3D8R CPE4R CPS4R CAX4R M3D4R PIPE21 PIPE31 S4 S4R S4RS S4RSW SAX1 T2D2 T3D2

Features tested

The small-displacement deformation theory.

Problem description

This verification test consists of a set of single-element models for each element type in analyses that use the small-displacement theory. All degrees of freedom are prescribed so that the results do not include any dynamic effects. Each element is subjected to all applicable fundamental modes of deformation. The total strains are large to show that the results are linear and remain unaffected by changes to the element's current configuration.

The material is linear elastic with a Young's modulus of 1.0×10^5, Poisson's ratio of .33, and density of 1000.

Results and discussion

All element types tested yield the appropriate results for their applicable fundamental modes of deformation. Results for the two-dimensional truss element are illustrated here.

There are two global modes of deformation for a two-dimensional truss: longitudinal and lateral. The longitudinal mode is driven by fixing one end of the truss and prescribing a longitudinal displacement at the other. The axial stresses in the truss element as a result of longitudinal deformation for both small-displacement theory (geometric nonlinearities are neglected) and large-displacement theory (geometric nonlinearities are considered in the step) are shown in *Figure 1*. As the strains become large, the results diverge because the large-displacement theory accounts for the thinning of the truss as it stretches. The global lateral mode is invoked by prescribing a lateral displacement at one end of the truss element while holding all other degrees of freedom fixed. Results for the lateral case are shown in *Figure 2*. The nonlinear geometric effect is accounted for only in the large-displacement analysis. The small-displacement analysis ignores the extension of the truss due to its rotation and, therefore, sees no extensional strain due to the prescribed lateral displacements.

Input files

lk_b21.inp	B21 elements.
lk_b22.inp	B22 elements.
lk_b31.inp	B31 elements.
lk_b32.inp	B32 elements.
lk_p21.inp	PIPE21 elements.
lk_p31.inp	PIPE31 elements.

lk_c3d8.inp	C3D8 elements.
lk_c3d8i.inp	C3D8I elements.
lk_c3d8_orient.inp	C3D8 elements with **ORIENTATION*.
lk_c3d8i_orient.inp	C3D8I elements with **ORIENTATION*.
lk_c3d8r.inp	C3D8R elements.
lk_c3d8r_orient.inp	C3D8R elements with **ORIENTATION*.
lk_cax4r.inp	CAX4R elements.
lk_cax4r_orient.inp	CAX4R elements with **ORIENTATION*.
lk_cpe4r.inp	CPE4R elements.
lk_cpe4r_orient.inp	CPE4R elements with **ORIENTATION*.
lk_cps4r.inp	CPS4R elements.
lk_cps4r_orient.inp	CPS4R elements with **ORIENTATION*.
lk_dashpota.inp	Dashpot elements.
lk_m3d4r.inp	M3D4R elements.
lk_m3d4r_orient.inp	M3D4R elements with **ORIENTATION*.
lk_s4.inp	S4 elements.
lk_s4_orient.inp	S4 elements with **ORIENTATION*.
lk_s4r.inp	S4R elements.
lk_s4r_orient.inp	S4R elements with **ORIENTATION*.
lk_s4r_gs.inp	S4R elements with **SHELL GENERAL SECTION*.
lk_s4r_gs_orient.inp	S4R elements with **SHELL GENERAL SECTION* and **ORIENTATION*.
lk_s4rs.inp	S4RS elements.
lk_s4rs_orient.inp	S4RS elements with **ORIENTATION*.
lk_s4rs_gs.inp	S4RS elements with **SHELL GENERAL SECTION*.
lk_s4rs_gs_orient.inp	S4RS elements with **SHELL GENERAL SECTION* and **ORIENTATION*.
lk_s4rsw.inp	S4RSW elements.
lk_s4rsw_orient.inp	S4RSW elements with **ORIENTATION*.
lk_s4rsw_gs.inp	S4RSW elements with **SHELL GENERAL SECTION*.
lk_s4rsw_gs_orient.inp	S4RSW elements with **SHELL GENERAL SECTION* and **ORIENTATION*.
lk_sax1.inp	SAX1 elements.
lk_sax1_gs.inp	SAX1 elements with **SHELL GENERAL SECTION*.
lk_springa.inp	Spring elements.
lk_t2d2.inp	Two-dimensional truss elements.
lk_t3d2.inp	Three-dimensional truss elements.

Figures

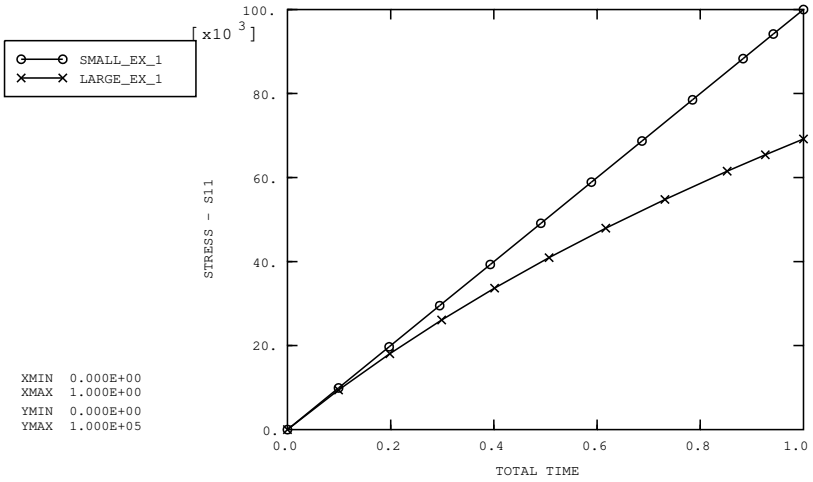


Figure 1: Axial stress comparison for the extensional mode.

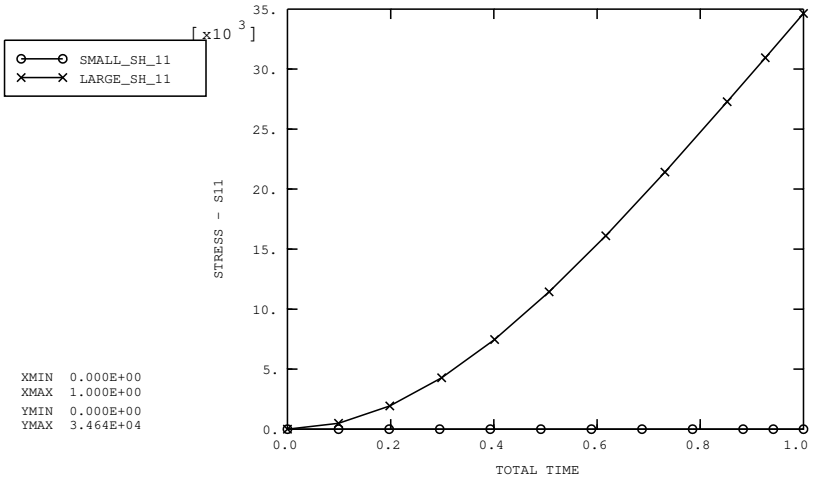


Figure 2: Axial stress comparison for the shear mode.

Mass scaling

Products: Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Various features of the fixed and variable mass scaling capabilities are tested. Most of the analyses consist of a set of reference elements that are unscaled and another set of test elements whose masses are scaled to equal those of the reference elements. The response of the test elements should be identical to that of the reference elements.

Verification of scaled mass matrices

Elements tested

B21 B22 B31 B32
C3D4 C3D6 C3D8R
CAX3 CAX4R
CPE3 CPE4R CPS3 CPS4R
M3D3 M3D4R MASS
R2D2 R3D3 R3D4 RAX2 ROTARYI
S3R S4R SAX1
T2D2 T3D2

Problem description

These problems verify that the element mass matrices are generated properly for every element type that can be scaled. Several element types are tested in each input file. For each element type an element pair consisting of a reference element and test element with identical geometries is defined. The material properties of each element pair are identical with the exception of the densities. The densities of the test elements are scaled with the FACTOR parameter so that in the analysis their element mass matrices are identical to those of the reference elements. Each element pair is subject to equivalent displacements (and rotations in the case of beams and shells) such that their response is dynamic. Rebars are included for every element type that permits the inclusion of rebar. Tests of membranes and shells are performed with and without nodal values specified with nodal thickness. Reaction forces for constrained nodes of each pair of elements are output for comparison purposes.

Results and discussion

Reaction force histories for nodes on each pair of test and reference elements are nearly identical. Slight differences exist because the bulk viscosity is based on the unscaled mass during the first increment. Every increment thereafter, the bulk viscosity is based on the scaled mass.

Input files

mscale_continuum.inp	Two-dimensional and three-dimensional continuum elements.
mscale_beamshell.inp	Two-dimensional and three-dimensional beams and shells.
mscale_special.inp	Elements with mass but no stable time increment.

Verification of mass scaling methods

Elements tested

M3D3 M3D4R S3R S4R SC6R SC8R SAX1
C3D4 CAX4R CPE3 CPE4R CPS3 CPS4R

Problem description

The various techniques are tested for fixed and variable mass scaling. In addition, the use of multiple mass scaling definitions is also tested. These problems consist of a set of reference elements and a set of test elements with identical geometries. The material properties of each set of reference and test elements are identical with the exception of the densities. The densities of the reference elements are scalar multiples of those of the test elements. The element stable time increment is assigned a value so that the masses of the test elements are scaled to exactly equal those of the reference elements. Displacement boundary conditions are used to deform each pair of elements; however, the deformation is minimal, so the element stable time increments are not affected significantly.

Results and discussion

Reaction force histories for nodes of each pair of the reference and test elements are nearly identical. Slight differences exist because the bulk viscosity is based on the unscaled mass during the first increment. Every increment thereafter, the bulk viscosity is based on the scaled mass. Furthermore, in cases in which variable mass scaling is specified, additional differences arise because of the continual scaling of the elements' masses throughout the step.

Input files

mscale_belowmin_fms.inp	**FIXED MASS SCALING*, TYPE=BELOW MIN.
mscale_belowmin_vms.inp	**VARIABLE MASS SCALING*, TYPE=BELOW MIN.
mscale_belowminfac.inp	**FIXED MASS SCALING*, TYPE=BELOW MIN with a mass scaling factor.
mscale_uniform_fms.inp	**FIXED MASS SCALING*, TYPE=UNIFORM.
mscale_uniform_vms.inp	**VARIABLE MASS SCALING*, TYPE=UNIFORM.
mscale_uniformfac.inp	**FIXED MASS SCALING*, TYPE=UNIFORM with a mass scaling factor.
mscale_setequaldt_fms.inp	**FIXED MASS SCALING*, TYPE=SET EQUAL DT.
mscale_setequaldt_vms.inp	**VARIABLE MASS SCALING*, TYPE=SET EQUAL DT.
mscale_setequaldtfac.inp	**FIXED MASS SCALING*, TYPE=SET EQUAL DT with a mass scaling factor.
mscale_multiuniform_fms.inp	Multiple uniform mass scaling definitions with **FIXED MASS SCALING*.
mscale_multiuniform_vms.inp	Multiple uniform mass scaling definitions with **VARIABLE MASS SCALING*.

Verification of FREQUENCY and NUMBER INTERVAL parameters

Elements tested

CPE4R

Problem description

The variable mass scaling is used throughout a step. In this problem a group of elements is stretched such that they experience severe distortions. The variable mass scaling is used to prevent the stable time increment from decreasing below a specified value. Two tests are performed in which the mass scaling is performed at specified increments and at specified time intervals during the step. The stable time increment and percent change in total mass are output to monitor the mass scaling of the model.

Results and discussion

Stable time increment histories show that they do not fall below the specified minimum. Time histories of the percent change in total mass show a continual increase, thereby verifying that mass is being scaled throughout the step.

Input files

mscale_frequency.inp	Scaling is performed at specified increments.
mscale_interval.inp	Scaling is performed at specified time intervals.

Mass scaling in a multiple step analysis

Elements tested
M3D4R

Problem description

Mass scaling definitions can be removed or propagated from step to step. Furthermore, the mass matrix of an element that has been scaled in a previous step can be propagated to a subsequent step or reinitialized to its original state. In this problem a combination of fixed and variable mass scaling definitions are defined over several steps to verify these mass scaling features for a multistep analysis. Reaction forces and the percent change in total mass of the model are output.

Results and discussion

Reaction force histories for nodes of the test and reference elements are identical. Examination of the reaction forces and the percent change in total mass of the model verifies that mass scaling definitions are propagated and removed correctly across steps. Mass matrices are also propagated and reinitialized correctly.

Input files

mscale_multistep.inp	Input data for this analysis.

Global and local mass scaling

Elements tested
CPE4R

Problem description

Mass scaling can be defined globally or locally on an element set basis. A local mass scaling definition will override a global mass scaling definition for an element, as verified in this problem.

Results and discussion

Mass scaling factor and element stable time increment histories verify that global mass scaling definitions are overwritten by local definitions for specified elements.

Input files

mscale_locglobal_fms.inp	Local and global **FIXED MASS SCALING* definitions.
mscale_locglobal_vms.inp	Local and global **VARIABLE MASS SCALING* definitions.

Mass scaling of rigid bodies

Elements tested

CPE4R C3D8R R2D2 R3D4 ROTARYI S4R

Problem description

Mass scaling of rigid elements or deformable elements defined as a rigid body can be performed. Techniques for scaling rigid bodies are limited because these elements do not have an associated stable time increment (*Mass Scaling*).

The following tests verify the use of the fixed and variable mass scaling with rigid bodies. These problems consist of a set of reference elements and two sets of test elements with identical geometries, as shown in *Figure 1*. Each element set consists of two independent bodies that come into contact: a fixed rigid surface and a body consisting of a combination of rigid and deformable elements. The material properties of the reference and test elements are identical with the exception of the densities. The densities of both sets of test elements are identical, but they are scaled for one set to equal those of the reference elements.

Initial velocities are applied in the vertical direction so that impact with the fixed rigid surfaces (elements 101, 111, and 121) occurs. Reaction forces at the reference nodes of the fixed rigid surfaces are output for comparison purposes.

Results and discussion

Vertical reaction force histories for the fixed rigid surfaces are nearly identical for the reference and scaled element sets, as shown in *Figure 2*. Very slight differences exist because the bulk viscosity is based on the unscaled mass during the first increment. Every increment thereafter, the bulk viscosity is based on the scaled mass.

Input files

mscale_rbod2d_fms1.inp	Two-dimensional continuum elements defined as a rigid body and attached to two-dimensional continuum elements with the **FIXED MASS SCALING* option applied only to the deformable elements.
mscale_rbod2d_fms2.inp	Two-dimensional continuum elements defined as a rigid body and attached to two-dimensional continuum elements with the **FIXED MASS SCALING* option applied to both deformable and rigid portions of the mesh.
mscale_rbod2d_fms3.inp	Two-dimensional continuum elements defined as a rigid body and attached to two-dimensional continuum elements with the **FIXED MASS SCALING*, TYPE=UNIFORM option applied to both the deformable and rigid portions of the mesh.
mscale_relem2d_fms1.inp	Rgid elements attached to two-dimensional continuum elements with the **FIXED MASS SCALING* option applied only to the deformable elements.

mscale_relem2d_fms2.inp	Rigid elements attached to two-dimensional continuum elements with the **FIXED MASS SCALING* option applied to both the deformable and rigid portions of the mesh.
mscale_relem2d_fms3.inp	Rigid elements attached to two-dimensional continuum elements with the **FIXED MASS SCALING*, TYPE=UNIFORM option applied to both the deformable and rigid portions of the mesh.
mscale_rbod2d_vms3.inp	Two-dimensional continuum elements defined as a rigid body and attached to two-dimensional continuum elements with the **VARIABLE MASS SCALING*, TYPE=UNIFORM option applied to both the deformable and rigid portions of the mesh.
mscale_relem2d_vms3.inp	Rigid elements attached to two-dimensional continuum elements with the **VARIABLE MASS SCALING*, TYPE=UNIFORM option applied to both the deformable and rigid portions of the mesh.
mscale_rbod3d_fms3.inp	**FIXED MASS SCALING*, TYPE=UNIFORM option applied to both the deformable and rigid portions of the mesh.
mscale_rbod3d_fms3_gcont.inp	**FIXED MASS SCALING*, TYPE=UNIFORM option applied to both the deformable and rigid portions of the mesh. Analysis using the general contact capability.
mscale_rbod3d_vms3.inp	**VARIABLE MASS SCALING*, TYPE=UNIFORM option applied to both the deformable and rigid portions of the mesh.
mscale_rbod3d_vms3_gcont.inp	**VARIABLE MASS SCALING*, TYPE=UNIFORM option applied to both the deformable and rigid portions of the mesh. Analysis using the general contact capability.
mscale_rbod3d_rotate.inp	Rotary inertia elements attached to a rigid surface.

Figures

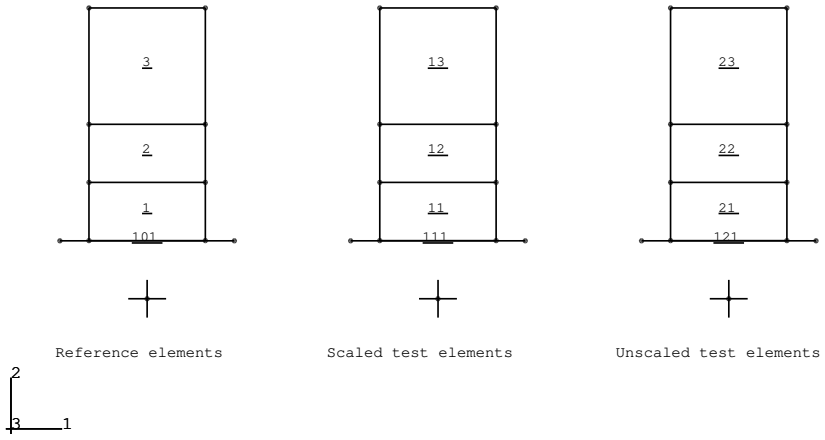


Figure 1: Mass scaling with rigid bodies.

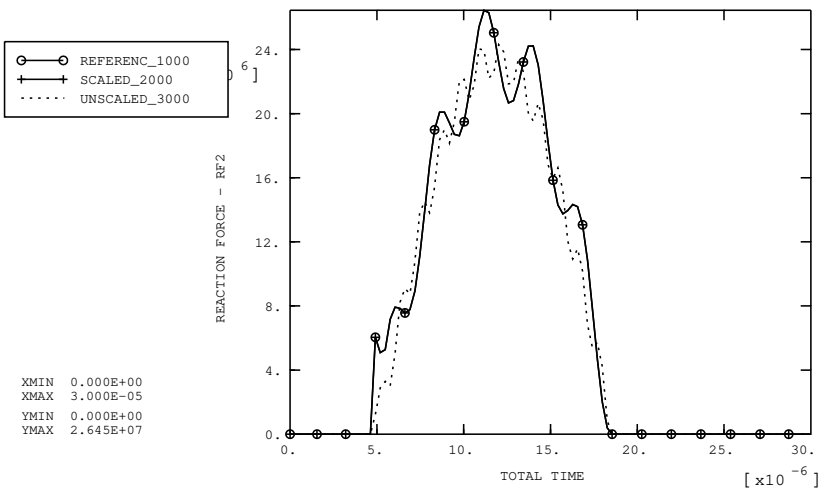


Figure 2: Comparison of vertical reaction forces on rigid surfaces.

Verification of mass scaling with kinematic contact

Elements tested

CPE4R C3D8R

Problem description

The contact forces resulting between two deformable bodies with kinematically enforced contact are functions of the masses at the nodes in contact, the magnitude of the time increment, and the penetration in the predicted configuration. These problems verify that the kinematic contact forces are calculated correctly when the densities for the contacting elements are scaled. Each problem consists of a set of reference elements and a set of test elements with identical geometries. Each set in turn consists of two independent bodies that come into contact. The material properties of the reference and test elements are identical with the exception of the densities. The densities of the test elements are scaled to equal those of the reference elements. Reaction force histories for nodes on the contacting bodies that are constrained are output for comparison purposes.

Results and discussion

Reaction force histories for nodes on each pair of test and reference elements are nearly identical. Slight differences exist because the bulk viscosity is based on the unscaled mass during the first increment. Every increment thereafter, the bulk viscosity is based on the scaled mass.

Input files

mscale_contact2d_fms.inp	CPE4R elements and **FIXED MASS SCALING*.
mscale_contact2d_vms.inp	CPE4R elements and **VARIABLE MASS SCALING*.
mscale_contact3d_fms.inp	C3D8R elements and **FIXED MASS SCALING*.
mscale_contact3d_vms.inp	C3D8R elements and **VARIABLE MASS SCALING*.

Verification of mass scaling with penalty contact

Elements tested

CPE4R C3D8R

Problem description

Nodal masses affect the penalty contact algorithm less directly than they affect the kinematic contact algorithm. Penalty contact forces depend on the penalty stiffness and the penetration in the current configuration. The penalty stiffnesses for contact between deformable surfaces are assigned automatically to a fraction of the elastic stiffness of the most compliant parent elements of the surfaces. Therefore, mass scaling does not influence the penalty contact forces between deformable surfaces for a given amount of penetration. However, nodal masses are factored into the effect of the penalty stiffness on the stable time increment. The problems from the previous subsection are repeated here with penalty enforcement of the contact constraints to verify that mass scaling is accounted for properly in the effect of the penalty stiffness on the stable time increment.

Results and discussion

The time increment decreases by about 4% during increments in which penalty contact forces are being transmitted. Some contact penetration can be observed in these tests, which is characteristic of the penalty contact method.

Input files

mscale_contactpnlty2d_fms.inp	CPE4R elements and **FIXED MASS SCALING*.
mscale_contactpnlty2d_vms.inp	CPE4R elements and **VARIABLE MASS SCALING*.
mscale_contactpnlty3d_fms.inp	C3D8R elements and **FIXED MASS SCALING*.
mscale_contact3d_fms_gcont.inp	C3D8R elements and **FIXED MASS SCALING* using the general contact capability.
mscale_contactpnlty3d_vms.inp	C3D8R elements and **VARIABLE MASS SCALING*.
mscale_contact3d_vms_gcont.inp	C3D8R elements and **VARIABLE MASS SCALING* using the general contact capability.

Crack propagation

In this section:

- *Crack propagation analysis*
- *Propagation of hydraulically driven fracture*
- *Hydraulically driven branching fracture*
- *Modeling slurry transport and placement in hydraulically driven fracture*

Crack propagation analysis

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

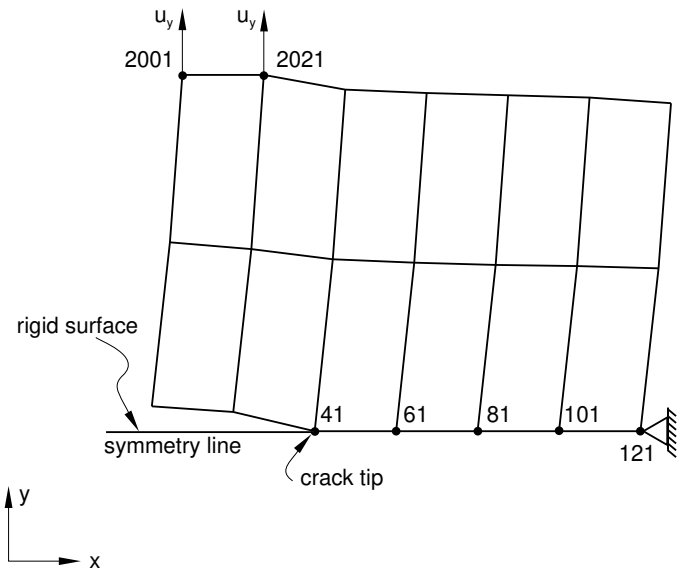
The tests in this section verify crack propagation between two surfaces that are initially partially bonded. They test the crack propagation capability from a single crack tip as well as multiple crack tips. All three fracture criteria (the critical stress criterion, the crack length versus time criterion, and the COD criterion) are verified.

Crack propagation from a single crack tip – edge notch plate

Elements tested

CPE4 CPE8

Problem description



In the symmetry model the top half of a single-edge notch plate is modeled with a mesh of 2×6 CPE4 elements. The lower surface of the bottom row of elements defines the secondary surface of the partially bonded contact pair, and the main surface is defined by an analytical rigid surface. The main surface also lies along the symmetry plane. Nonzero displacement boundary conditions are applied at two nodes remote from the symmetry plane. The time for bond failure and the position of the node at which the bond failure occurs (obtained from *pdebnods.inp*) are used to give the crack length versus time data in *pdebcrgr.inp*. The crack opening displacement at a distance behind the crack tip (obtained from *pdebnods.inp*) is used to specify the data for the COD criterion in *pdebcods.inp*. The stresses at a distance ahead of the crack tip (obtained from *pdebnods.inp*) are used to specify the data in *pdebnodsd.inp*. The time from *pdebnods.inp* is also used to set the time period for each step in *pdebchck.inp*.

The complete mesh is analyzed in *pdebnods2.inp*, *pdebcrgr2.inp*, and *pdebcods2.inp*.

Input files *pdebnodnlg.inp*, *pdebcrgnlg.inp*, and *pdebcodnlg.inp* consider finite deformation and finite sliding. The crack length versus time data for *pdebcrgnlg.inp* and the COD data for *pdebcodnlg.inp* are obtained from *pdebnodnlg.inp*.

Results and discussion

The time at bond failure, the remaining fraction of the stress at debonding, the remaining debond stress, and all element stresses and strains must be the same for corresponding increments of tests *pdebnods.inp*, *pdebcrgr.inp*, and *pdebcods.inp*. At the total time corresponding to the end of each step in *pdebchck.inp*, the stresses and strains in the continuum elements are the same for all three tests. The same results are obtained for the models analyzed in *pdebnods2.inp*, *pdebcrgr2.inp*, and *pdebcods2.inp*.

The results obtained from *pdebnodnlg.inp* are compared with that of *pdebcodnlg.inp* and *pdebcrgnlg.inp*. The time at bond failure, the debond stress at failure, and the element stresses and strains are the same at the corresponding times.

Input files

The following problems test the crack propagation capability for an edge crack notch plate with symmetry conditions taken into account:

pdebnods.inp	Tests crack propagation using a critical stress criterion. The distance ahead of the crack tip at which the critical stress is evaluated is set to zero.
pdebcrgr.inp	Tests crack propagation capability by using the crack length versus time criterion.
pdebcods.inp	Tests crack propagation capability by using the COD criterion.
pdebchck.inp	Checks this procedure without using any contact surface definitions by simulating the debonding by **BOUNDARY*, OP=NEW with multiple steps.
pdebnodsd.inp	Tests crack propagation capability by considering the critical stress at a distance ahead of the crack tip. The distance ahead of the crack tip at which the critical stress is evaluated is varied from step to step.

The following problems simulate the complete model:

pdebnods2.inp	Tests crack propagation capability by using a critical stress criterion. The distance ahead of the crack tip at which the critical stress is evaluated is set to zero.
pdebcrgr2.inp	Tests crack propagation capability by using the crack length versus time criterion.
pdebcods2.inp	Tests crack propagation capability by using the COD criterion.

The following verification tests involve finite deformation and finite sliding:

pdebnodnlg.inp	Tests crack propagation capability by using a critical stress criterion. The distance ahead of the crack tip at which the critical stress is evaluated is set to zero.
pdebcrgnlg.inp	Tests crack propagation capability by using the crack length versus time criterion.
pdebcodnlg.inp	Tests crack propagation capability by using the COD criterion.

The following files simulate crack propagation in the symmetry model using 8-node elements:

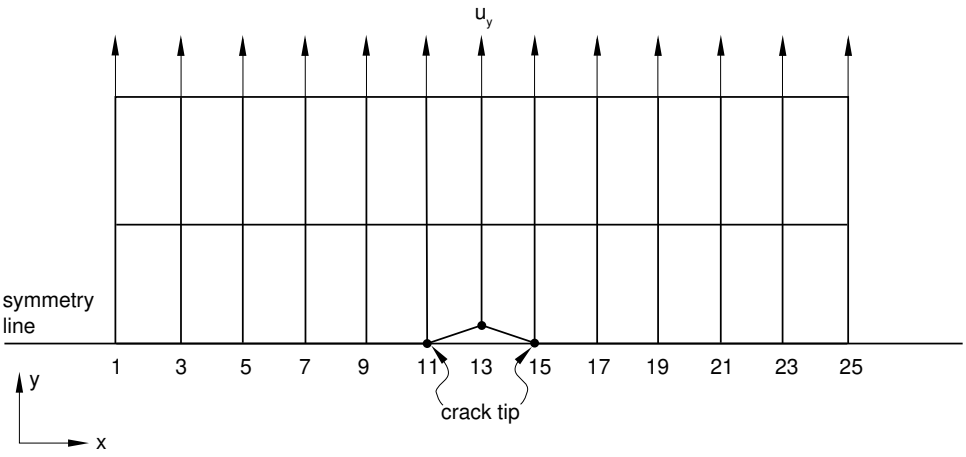
pdebnods8.inp	Tests crack propagation capability by using a critical stress criterion. The distance ahead of the crack tip at which the critical stress is evaluated is set to zero.
pdebcrgr8.inp	Tests crack propagation capability by using the crack length versus time criterion.
pdebcods8.inp	Tests crack propagation capability by using the COD criterion.

Crack propagation from multiple crack tips – center crack plate

Elements tested

CPE4

Problem description



The top half of a center cracked plate is modeled with a mesh of 2 × 12 CPE4 elements. The lower surface of the bottom row of elements is used to define the secondary surface of the partially bonded contact pair, and the main surface is defined by an analytical rigid surface. The main surface also lies on the symmetry plane. Nonzero displacement boundary conditions are applied on the top row of nodes.

The time for bond failure and the position of the node at which the bond failure occurs (obtained from *pdebnodcc1.inp*) are used to give the crack length versus time data in *pdebcrgcc1.inp*. The reference point for the crack length versus time criterion is defined such that the crack propagation occurs simultaneously from both the crack tips.

The crack opening displacement at a distance behind the crack tip (obtained from *pdebcodcc1.inp*) is used to specify the data for the COD criterion in *pdebcodcc1.inp*.

The complete mesh is analyzed in *pdebnodcc2.inp*, *pdebcodcc2.inp*, and *pdebcrgcc2.inp*.

Results and discussion

The time to bond failure and the debond stress at the time of bond failure are the same in all the tests. The stresses and strains in the elements are the same at a given time in all the tests.

Input files

pdebnodcc1.inp	Tests crack propagation using a critical stress criterion. The distance ahead of the crack tip at which the critical stress is evaluated is set to zero.
pdebcrgcc1.inp	Tests crack propagation capability by using the crack length versus time criterion.
pdebcodcc1.inp	Tests crack propagation capability by using the COD criterion.

The following input files simulate the complete model:

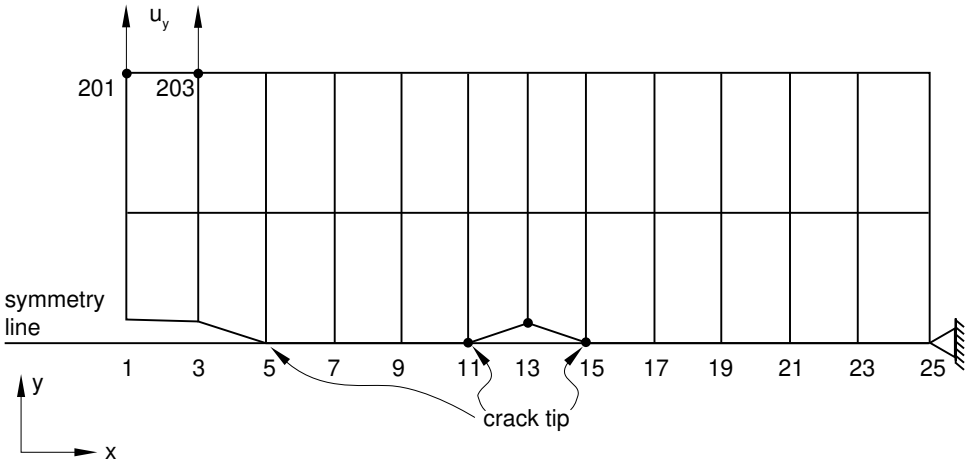
pdebnodcc2.inp	Tests crack propagation using a critical stress criterion. The distance ahead of the crack tip at which the critical stress is evaluated is set to zero.
pdebcrgcc2.inp	Tests crack propagation capability by using the crack length versus time criterion.
pdebcodcc2.inp	Tests crack propagation capability by using the COD criterion.

Crack coalescence

Elements tested

CPE4

Problem description



The top half of a plate that consists of an edge crack and a center crack is modeled with a mesh consisting of 2 × 12 CPE4 elements. The bottom surface of the lower row of elements is used to define the secondary surface of the initially partially bonded contact pair. The main surface of the contact pair is defined by an analytical rigid surface and also lies along the symmetry plane. Nonzero displacement boundary conditions are applied at two nodes remote from the bonded plane, as shown in the figure.

The complete mesh is analyzed in *pdebcrgco2.inp* and *pdebcodco2.inp*.

Results and discussion

The time to bond failure and the debond stress at the time of bond failure are the same in all the tests. The stresses and strains in the elements are the same at a given time in all the tests.

Input files

The following series of tests is used to demonstrate crack propagation and coalescence of two cracks:

pdebcrgco1.inp	Tests crack coalescence by using the crack length versus time criterion.
pdebcodco1.inp	Tests crack coalescence by using the COD criterion.

The following input files simulate the complete model:

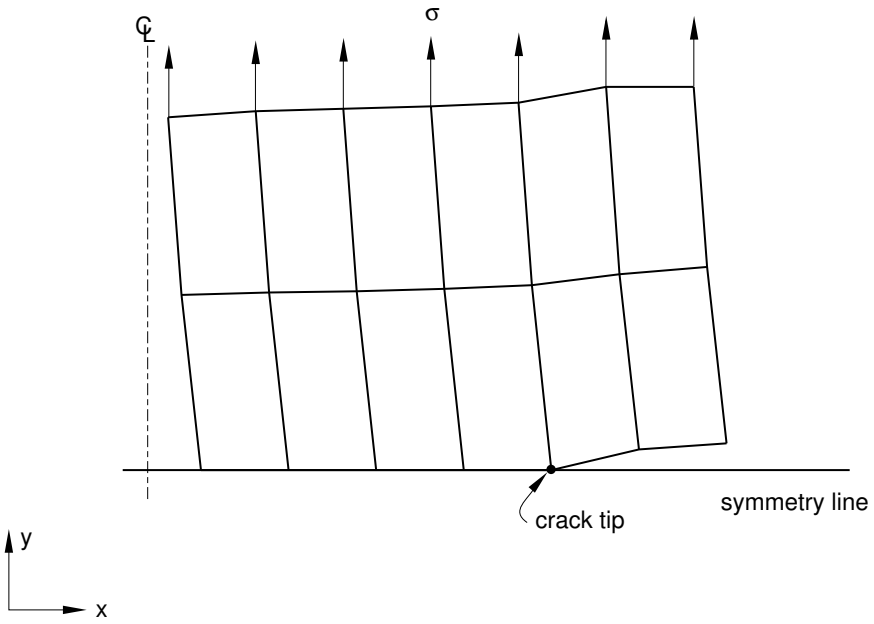
pdebcrgco2.inp	Tests crack coalescence by using the crack length versus time criterion.
pdebcodco2.inp	Tests crack coalescence by using the COD criterion.

Crack propagation analysis with axisymmetric elements

Elements tested

CAX4

Problem description



The problem of a round bar with an external notch (crack) subjected to tensile loading is analyzed as an axisymmetric case. Only the top half is modeled in *pdebnodax1.inp*, *pdebcrgax1.inp*, and *pdebcodax1.inp*. The mesh consists of 2×6 CAX4 elements. The lower surface of the bottom row of elements is used to define the secondary surface, while the main surface is defined by an analytical rigid surface. A far-field load $\boldsymbol{\sigma}$ is applied.

Input file *pdebnodax1.inp* uses the critical stress criterion for crack propagation analysis. The crack length versus time data for *pdebcrgax1.inp* and the crack opening displacement versus cumulative incremental crack length for *pdebcodax1.inp* are obtained from *pdebnodax1.inp*.

The complete mesh is analyzed in *pdebnodax2.inp*, *pdebcrgax2.inp*, and *pdebcodax2.inp*.

Results and discussion

The time to bond failure and the debond stress at the time of bond failure are the same in all the tests. The stresses and strains in the elements are the same at a given time in all the tests.

Input files

The following tests are used to verify the crack propagation capability for axisymmetric elements:

pdebnodax1.inp	Tests crack propagation using a critical stress criterion. The distance ahead of the crack tip at which the critical stress is evaluated is set to zero.
pdebcrgax1.inp	Tests crack propagation by using the crack length versus time criterion.
pdebcodax1.inp	Tests crack propagation by using the COD criterion.

The following input files simulate the complete model:

pdebnodax2.inp	Tests crack propagation using a critical stress criterion. The distance ahead of the crack tip at which the critical stress is evaluated is set to zero.
pdebcrgax2.inp	Tests crack propagation by using the crack length versus time criterion.
pdebcodax2.inp	Tests crack propagation by using the COD criterion.

Miscellaneous test

Elements tested

CPE4

Problem description

This input file tests for the output of the debond variables.

Results and discussion

The debond variables are identical to those obtained in the original analysis.

Input files

pdebnodsps.inp	Tests the **POST OUTPUT* option. The restart file from pdebnods.inp is needed to run this input file.

Propagation of hydraulically driven fracture

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

The tests in this section verify the hydraulically driven crack propagation in a permeable porous medium.

Elements tested

COH2D4P COH3D8P

Problem description

The plane strain model consists of a half-circle plate with one layer of pore pressure cohesive elements inserted in the middle. A two-step transient soil consolidation analysis is performed, and crack propagation is developed along the layer of cohesive elements. In the first step a linearly increased flow rate is initially applied at the crack mouth located at the left side of cohesive element layer, after which the flow rate will be kept constant for the rest of time. In the second step the injection of the fluid is terminated and the built-up pore pressure in fracture is allowed to bleed off into the medium. The same plane strain problem is also modeled using one layer of three-dimensional elements.

Results and discussion

In the first step steady crack propagation can be observed, with pressure built up gradually inside the developed crack. In the second step the crack was eventually closed, since the built-up pressure bled off into the medium. The same behaviors can be observed in the two-dimensional models and the three-dimensional model.

Input files

hydrfract-coh2d4p.inp	Plane strain model using COH2D4P and CPE4RP elements.
hydrfract-coh3d8p.inp	Plane strain model using COH3D8P and C3D8RP elements.
hydrfract-coh2d4p_uleakoff.inp	The same model as hydrfract-coh2d4p.inp, but with fluid leak-off specified using user subroutine *UFLUIDLEAKOFF*.
hydrfract-coh2d4p_uleakoff.f	User subroutine *UFLUIDLEAKOFF* used for hydrfract-coh2d4p_uleakoff.inp.

Hydraulically driven branching fracture

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

The test in this section verifies the hydraulically driven branching fracture propagation using the pore pressure cohesive elements with the fluid transition from Darcy flow to Poiseuille flow.

Elements tested

COD2D4P

Problem description

The test verifies the crack branching functionality by using the pore pressure cohesive elements with the fluid transition from Darcy flow to Poiseuille flow. The model simulates the hydraulic fracture growth with existing cracks. There are three branches of cohesive zones: a vertical branch and two horizontal branches. The two horizontal branches represent preexisting cracks, and the vertical branch simulates progressive fracture by hydraulic injection. The material strength is weakened for the horizontal branches to simulate the preexisting cracks. Confining pressure is prescribed around the whole domain. Contact with friction is prescribed to the cohesive zone top and bottom surfaces to study the role of friction on the crack propagation into preexisting fracture.

Results and discussion

The results match the results presented in *Zhang and Jeffrey (2006)*. When the fracture propagates into the horizontal branches, the crack changes mode from tension into shearing, the injection pressure increases, and the crack width decreases. The friction coefficient value between the top and bottom surfaces of the cohesive zones plays an important role in how the fluid gets into the preexisting fractures and how the horizontal fracture propagates.

Input files

ac-nogap5c.inp	Plane strain model using COD2D4P and CPE4 elements.

References

- Zhang, X., and R. G. Jeffrey, “The Role of Friction and Secondary Flaws on Deflection and Re-initiation of Hydraulic Fractures at Orthogonal Pre-existing Fractures,” *Geophysical Journal International*, no. 166, pp. 1454–1465, 2006.

Modeling slurry transport and placement in hydraulically driven fracture

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

The tests in this section verify the slurry transport and placement in hydraulically driven crack propagation in a permeable porous medium.

Elements tested

COD2D4C COD3D8C CPE4P C3D8P

Problem description

The problem consists of a rectangular block 20 m long, 2.5 m wide, and 1 m thick. Both XFEM and cohesive elements are used to model the fracture with slurry transport and placement in this analysis. Two different models verify different aspects of the slurry transport and placement:

- Slot test to verify slurry advection and settling in fixed-opening fractures
- Hydraulic fracture test to verify the crack profile and slurry advection

In the slot test, the dimensions of the slot channel are 1 m long, 0.2 m high, and 0.5 mm wide. The injection rate is 0.0004 m^3/h. The injected proppant has the following properties:

$$C_V^{inlet} = 0.2$$

$$\rho_w = 1000\,kg/m^3, \rho_s = 1250\,kg/m^3$$

The diameter of the proppant particles is assumed to be $D_p = 0.01\ mm.$

In the hydraulic fracture test, a small initial crack is specified at one end of the specimen. A two-step analysis is performed to simulate the hydraulic fracture propagation. First, a geostatic step is performed to establish the initial equilibrium state. The next step represents the hydraulic fracture stage, where the slurry with a proppant concentration of 0.6 is injected into the well with an injection rate of 50 m^3/h. Similar studies are conducted through applied pore pressure of 120 MPa. The element adjacent to the wellbore is defined as initially open to permit entry of the slurry. The duration of this stage is 0.1 s, and during this time period both the applied volumetric flow rate (or the applied pore pressure) and concentration are ramped up linearly. Additional tests using the same model exercise the following:

- leakoff with a dynamic filtration model;
- user subroutine to set crack opening offset for contact of crack surfaces;
- minimum ratio of the crack opening to the particle size, necessary to activate the slurry calculations; and
- deactivate the slurry flow in an analysis.

Results and discussion

The results for the XFEM and cohesive approaches are compared to each other, as well as with available analytical solutions obtained by solving a one-dimensional slurry flow continuity equation (accounting for leakoff). Good agreement is obtained in all cases.

Figures

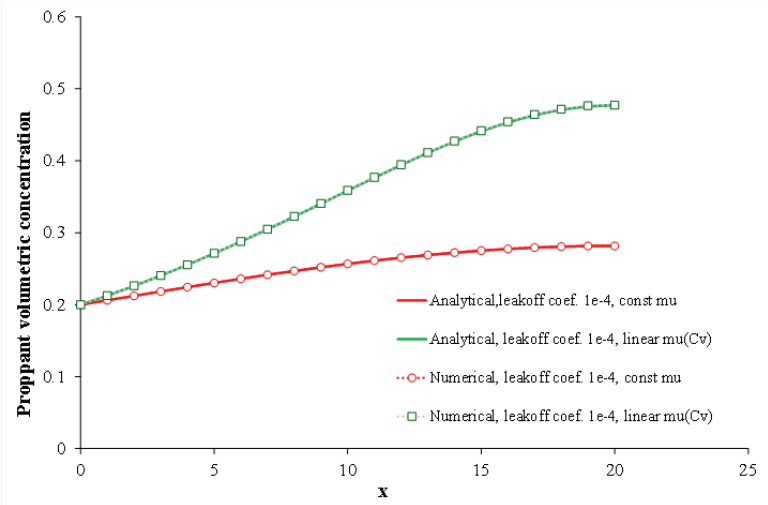


Figure 1: A comparison of the slurry concentration profile obtained using cohesive elements with those obtained analytically.

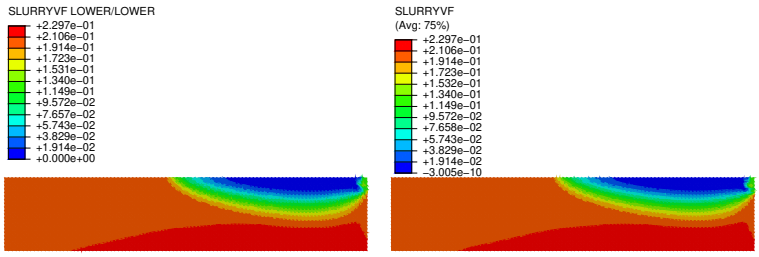


Figure 2: A comparison of slurry concentration profiles for the slot test obtained using XFEM (right) and cohesive elements (left).

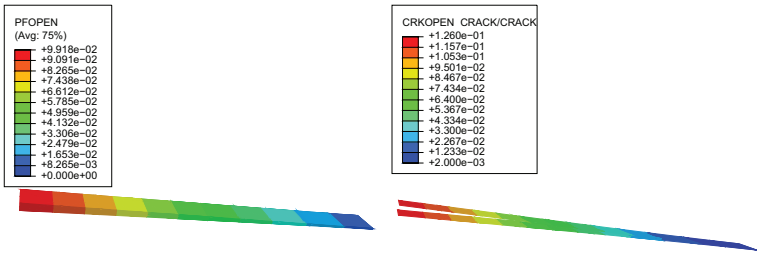


Figure 3: A comparison of crack profile for the hydraulic fracture test obtained using XFEM (right) and cohesive elements (left).

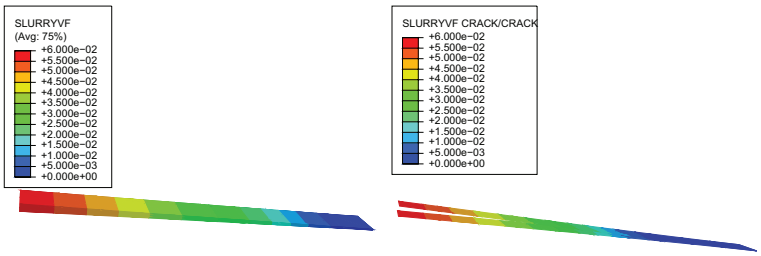


Figure 4: A comparison of slurry concentration profile for the hydraulic fracture test obtained using XFEM (right) and cohesive elements (left).

Substructuring

In this section:

- *Substructure rotation, mirroring, transformation, and constraints*
- *Substructure recovery with The nodal Transformation*
- *Degenerated elements within a substructure*
- *Substructure load case with centrifugal loads*
- *Substructure load case with nodal pressure loads*
- *Thermal-stress analysis with substructures*
- *Substructure preload history*
- *Substructure removal*
- *Substructure damping*
- *Substructures with rebar*
- *Frequency extraction for substructures*
- *Substructures with large rotations*
- *Coupled structural-acoustic analysis with substructures*

Substructure rotation, mirroring, transformation, and constraints

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Substructure rotation, transformation, and kinematic constraints

Features tested

Rotation of a substructure and the recovery of nodal and element variables, material directions, and integration point coordinates. The equation constraints, multi-point constraints, and nodal transformation are verified.

Problem description

A rectangular substructure of length 10.0 and thickness and width 1.0 is formed and subjected to a pressure load of −200.0 on one end. The substructure is rotated 30° and fixed at the end opposite to the pressure load. A 2×5 mesh is used for solid and shell elements, and a 10-element mesh is used for beam elements.

A second mesh is defined in the rotated position and is loaded in the same manner as the first mesh, but without using substructures. The displacements, strains, and stresses, as well as the integration point coordinates and the material directions, should be identical for the elements within the substructure and the elements defined without using a substructure. The substructure stresses and strains are reported in the global system for continuum elements. In all other cases the substructure stresses and strains are reported in the rotated system.

The use of equation constraints, multi-purpose constraints, and nodal transformation is tested on the substructure comprised of CPE4 elements. The nodal transformation is tested both in the usage and in the creation level. Three levels of substructures are created for this particular analysis. The lowest level is a 2×1 mesh of CPE4 elements. The next level comprises two of the first-level substructures, and the third level is the actual structure. The use of unsorted retained degrees of freedom is tested during the creation levels.

Results and discussion

All results in the substructure are identical to the results in the regular mesh and are within 0.05% of the analytical uniaxial solution.

Input files

psupsol1.inp	CPE4 elements without **ORIENTATION*.
psupsol1_gen.inp	Substructure generation file referenced in psupsol1.inp.
psupsol1or.inp	CPE4 elements with **ORIENTATION*.
psupsol1or_gen.inp	Substructure generation file referenced in psupsol1or.inp.
psupsol1mi.inp	CPE4 elements with **TRANSFORM*, **MPC*, **EQUATION*, and unsorted retained DOFs.
psupsol1mi_gen1.inp	Substructure generation file referenced in psupsol1mi.inp.
psupsol1mi_gen2.inp	Substructure generation file referenced in psupsol1mi.inp.
psupsol1mi_gen3.inp	Substructure generation file referenced in psupsol1mi.inp.
psupcontshl.inp	SC8R elements with **SHELL SECTION* and without **ORIENTATION*.

psupcontshl_gen.inp	Substructure generation file referenced in psupcontshl.inp.
psupshl1.inp	S4R elements with **SHELL SECTION* and without **ORIENTATION*.
psupshl1_gen.inp	Substructure generation file referenced in psupshl1.inp.
psupshl1or.inp	S4R elements with **SHELL SECTION* and **ORIENTATION*.
psupshl1or_gen.inp	Substructure generation file referenced in psupshl1or.inp.
psupsfl1.inp	S4 elements with **SHELL SECTION* and without **ORIENTATION*.
psupsfl1_gen.inp	Substructure generation file referenced in psupsfl1.inp.
psupsfl1or.inp	S4 elements with **SHELL SECTION* and **ORIENTATION*.
psupsfl1or_gen.inp	Substructure generation file referenced in psupsfl1or.inp.
psupshl2.inp	S4R elements with **SHELL GENERAL SECTION* and without **ORIENTATION*.
psupshl2_gen.inp	Substructure generation file referenced in psupshl2.inp.
psupshl2or.inp	S4R elements with **SHELL GENERAL SECTION* and **ORIENTATION*.
psupshl2or_gen.inp	Substructure generation file referenced in psupshl2or.inp.
psupsfl2.inp	S4 elements with **SHELL GENERAL SECTION* and without **ORIENTATION*.
psupsfl2_gen.inp	Substructure generation file referenced in psupsfl2.inp.
psupsfl2or.inp	S4 elements with **SHELL GENERAL SECTION* and **ORIENTATION*.
psupsfl2or_gen.inp	Substructure generation file referenced in psupsfl2or.inp.
psupbm1.inp	B31 elements with **BEAM SECTION*.
psupbm1_gen.inp	Substructure generation file referenced in psupbm1.inp.
psupbm2.inp	B31 elements with **BEAM GENERAL SECTION*.
psupbm2_gen.inp	Substructure generation file referenced in psupbm2.inp.

Substructure rotation and mirroring

Features tested

Translation, rotation, and mirroring of multilevel substructures and the recovery of nodal and element variables. These features are tested on two different models, a hemispherical shell and a simple hoist model. The hemispherical shell model is the same as that described in *LE3: Hemispherical shell with point loads*.

Problem description

Two models are discussed below.

Hemispherical shell model

The mesh for the hemispherical shell problem in *LE3: Hemispherical shell with point loads* consists of S4R5 elements. In that example one-eighth of the sphere is modeled. In this example the mesh is divided into three equal parts, as shown in *Figure 1*, with each part modeled with a 4×4 mesh of shell elements.

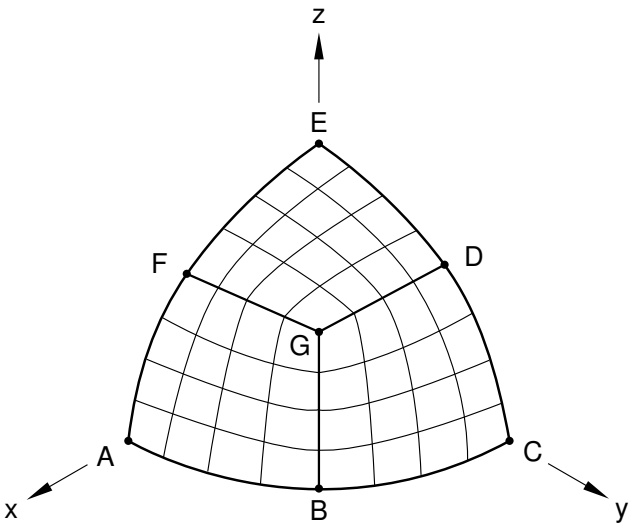


Figure 1: Hemispherical shell model.

One of the three parts is defined (A - B - G - F), and a substructure is created. One-eighth of the sphere is then obtained by mirroring the substructure over lines F - G and G - B, respectively. The results from *LE3: Hemispherical shell with point loads* are reproduced.

In the second example one-quarter of the sphere is modeled by using this substructure twice, the second time rotating it 90° around the z-axis.

In the third example one-half of the sphere is modeled by using the new substructure twice, the second time mirroring it in the x–z plane.

Overhead hoist model

The overhead hoist shown in *Figure 2* is used to test translation, rotation, and mirroring of a multilevel substructure. The hoist is a simple pin-jointed beam and truss model that is constrained at the left-hand end and mounted on rollers at the right-hand end. The members can rotate freely at the joints. Each member is 1 m in length and 5 mm in diameter. The structure is subjected to a 10 kN load at the center. The Young's modulus and Poisson's ratio of the members are taken to be 200 GPa and 0.3, respectively. The structure is modeled using seven T2D2 elements, one element for every member.

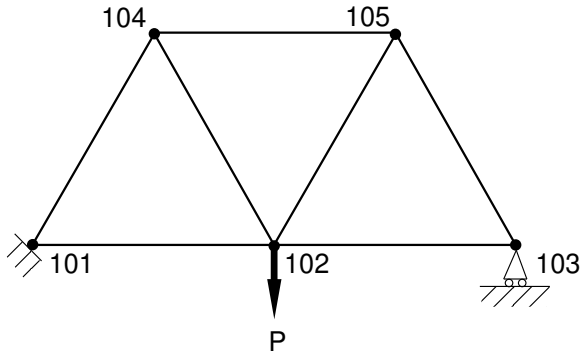


Figure 2: Overhead hoist model.

The horizontal member connected to the fixed end is used to form the first-level substructure. The second-level substructure representing the triangular section of the hoist is then formed by rotating and translating the first-level substructure. The third-level substructure representing the actual structure is created by mirroring or translating the lower-level substructures.

An independent model of the structure using regular T2D2 elements is also created to verify the results obtained.

Results and discussion

The results for each model are discussed below.

Hemispherical shell model

All element output is in the local directions defined during the substructure formation.

Mesh	Element	Sec. pt.	σ_{11}	σ_{22}	σ_{12}
psuplev1	3000 < 1	1	–377.	–2148.	2581.
psuplev1	3000 < 2	1	377.	2148.	–2581.
psuplev1	3000 < 3	1	1.	1.	700.
psuplev2	3000 < 1 < 101	1	–377.	–2148.	2581.
psuplev2	3000 < 2 < 101	1	378.	2149.	–2580.
psuplev2	3000 < 3 < 101	1	2.	1.	699.
psuplev2	3000 < 1 < 102	1	377.	2147.	–2581.
psuplev2	3000 < 2 < 102	1	–378.	–2149.	2580.
psuplev2	3000 < 3 < 102	1	–2.	–1.	–700.
psuplev3	3000 < 1 < 101 < 1001	1	–377.	–2148.	2581.
psuplev3	3000 < 3 < 101 < 1001	1	2.	1.	699.
psuplev3	3000 < 1 < 102 < 1001	1	377.	2147.	–2581.
psuplev3	3000 < 1 < 101 < 1002	1	–377.	–2148.	2581.
psuplev3	3000 < 2 < 101 < 1002	1	378.	2149.	–2580.
psuplev3	3000 < 1 < 102 < 1002	1	377.	2147.	–2581.
psuplev3	3000 < 2 < 102 < 1002	1	–378.	–2149.	2580.
psuplev3	3000 < 3 < 102 < 1002	1	–2.	–1.	–700.

Overhead hoist model

The results obtained with the multilevel substructure analysis are identical to the results obtained with the regular element model. The nodal displacements, u_{Reg} for the regular element model and u_{Sup} for the substructure model, are given in the following table:

Node	u_{1Sup}	u_{1Reg}	u_{2Sup}	u_{2Reg}
102	7.456E–04	7.456E–04	–4.735E–03	–4.735E–03
103	1.491E–03	1.491E–03	0.000E–00	0.000E–00
104	1.491E–03	1.491E–03	–2.583E–03	–2.583E–03
105	0.000E–00	0.000E–00	–2.583E–03	–2.583E–03

Input files

psuplev1.inp	This is an analysis of one-eighth of a sphere.
psuplev1_gen.inp	Forms the first-level substructures; referenced in analysis psuplev1.inp.

psuplev2.inp	This performs the analysis of one-fourth of a sphere by using two of the substructures, the second one rotated 90°.
psuplev2_gen.inp	Forms the second-level substructures by using three of the first-level substructures: one in the original geometric location and two by mirroring the element; referenced in analysis psuplev2.inp.
psuplev3.inp	This performs the global analysis by using two copies of the substructure: one in the original position and one by mirroring.
psuplev3_gen.inp	Forms the third-level substructures by using two of the second-level substructures: one in the original geometric location and the other by rotating the substructure 90°; referenced in analysis psuplev3.inp.
psuphoi1.inp	Analysis of a simple overhead hoist model using substructures.
psuphoi1_gen1.inp	Substructure generation file referenced in psuphoi1.inp.
psuphoi1_gen2.inp	Substructure generation file referenced in psuphoi1.inp.
psuphoi2.inp	Analysis of overhead hoist model without substructures.

Substructure rotation activating nonretained DOF

Features tested

Substructure rotation that activates degrees of freedom that were not retained during substructure generation.

Problem description

A substructure is defined along the global x-axis by retaining the x-displacement degree of freedom at both nodes of a T2D2 element. The substructure property definition is used to rotate the substructure 45° in the x–y plane. One end of the substructure is fixed, whereas displacement boundary conditions corresponding to axial tension are applied at the free end.

Results and discussion

The results from the substructure analysis exactly match the results that were obtained when substructures were not used. The degrees of freedom that were not retained during substructure generation are activated properly by the use of the substructure property definition.

Input files

psuptr1.inp	Uses one T2D2 element.
psuptr1_gen.inp	Substructure generation file referenced in psuptr1.inp.

Including deformable elements that are declared as rigid

Features tested

Inclusion of deformable elements that are declared as rigid during substructure generation and subsequent usage is verified.

Problem description

The use of deformable elements that are declared as rigid is tested at the substructure generation level and at the usage level. The substructure mesh consists of 10 beam elements with one of the elements declared as rigid. A pressure load of −200.0 is applied on one end. The substructure is rotated 30° and fixed at the end opposite to

the pressure load. A second mesh is defined in the rotated position and loaded in the same manner as the substructure mesh. This mesh consists of beam elements with one of the elements declared as rigid. Substructures are not included in this mesh. The displacements, strains, and stresses should be identical for the elements within the substructure and the elements defined without using a substructure.

Results and discussion

All results in the substructure are identical to the results in the regular mesh.

Input files

psupbm11.inp	A deformable element is declared as rigid at the substructure generation level and at the usage level.
psupbm11_gen.inp	Substructure generation file referenced in psupbm11.inp.

Substructure recovery with The nodal Transformation

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

CPE4 CPEG4

Features tested

The nodal transformation is used at the substructure usage level for a retained node of a substructure. The recovery of variables inside the substructure should not be affected by the transformation.

Problem description

In all the analyses a local coordinate system is defined at two adjacent nodes of a 1.0×1.0 planar substructure. Displacement boundary conditions are prescribed at these nodes such that the net effect is pure extension in the global x-direction. In the first and third analyses a local Cartesian coordinate system is defined at an angle of 45° with the global Cartesian system; in the second analysis a local cylindrical coordinate system is defined such that the axis of the cylindrical system is parallel to the global z-axis; and in the fourth analysis a local spherical coordinate system is defined such that the polar axis is parallel to the global z-axis. The center of the spherical system is defined such that the radial direction at all the nodes coincides with the global x-direction.

Results and discussion

Results on the substructure level for all four analyses are identical to analytical results in which a stress of σ_{11}=424264. develops in the direction of extension.

Input files

psuptrn1.inp	Uses CPE4 elements with a rectangular Cartesian nodal coordinate system.
psuptrn1_gen.inp	Substructure generation file referenced in the analysis psuptrn1.inp.
psuptrn2.inp	Uses CPE4 elements with a cylindrical local coordinate system.
psuptrn2_gen.inp	Substructure generation file referenced in the analysis psuptrn2.inp.
psuptrn3.inp	Uses CPEG4 elements with a rectangular Cartesian nodal coordinate system.
psuptrn3_gen.inp	Substructure generation file referenced in the analysis psuptrn3.inp.
psuptrn4.inp	Uses CPEG4 elements with a spherical local coordinate system.
psuptrn4_gen.inp	Substructure generation file referenced in the analysis psuptrn4.inp.

Degenerated elements within a substructure

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

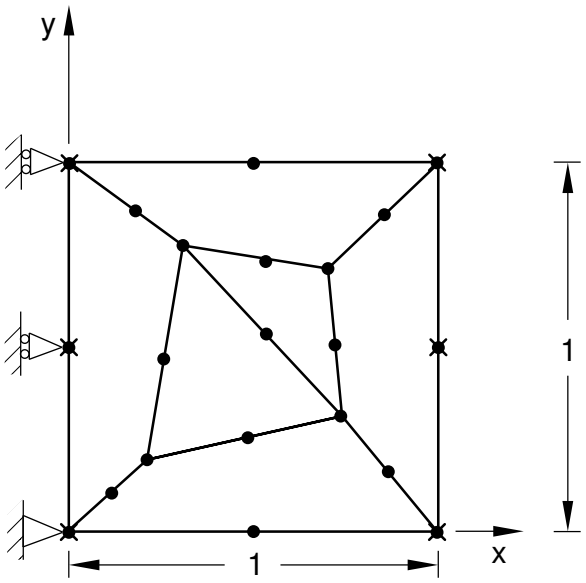
CPS8

Features tested

The ability to incorporate degenerated elements into a substructure definition is tested.

Problem description

A planar substructure is formed from standard CPS8 elements and CPS8 elements degenerated into 6-node triangles. Both displacement degrees of freedom are retained at selected nodes on the substructure that is constrained as depicted below, and a displacement of 0.2 in the x-direction is prescribed for the three nodes along x=1.0.



Results and discussion

The results are identical to the analytical results where a stress of 6×10^6 develops in the direction of extension.

Input files

psupdgn1.inp	Uses CPS8 and degenerated CPS8 elements.
psupdgn1_gen.inp	Substructure generation file referenced in the analysis psupdgn1.inp.

Substructure load case with centrifugal loads

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

CPE8R

Features tested

The ability to define a substructure load case with centrifugal loads and apply it on the usage level is tested.

Problem description

A substructure is defined from a CPE8R element, and a load vector representing centrifugal loading is defined via a substructure load case using distributed centrifugal loads (load types CENT and CENTRIF). Four such substructures are then used to model one-quarter of a rotating annular disk in the x–y plane.

Results and discussion

The results are identical to those from an analysis without substructures. The displacement of the inner and outer radius at all points on the quarter annulus is 0.1833 and 0.2388, respectively. Both displacements are within 2% of the analytical results.

Input files

psupsld1.inp	Uses CPE8R elements with CENT loading.
psupsld1_gen.inp	Substructure generation file referenced in the analysis psupsld1.inp.
psupsld2.inp	Uses CPE8R elements with CENTRIF loading.
psupsld2_gen.inp	Substructure generation file referenced in the analysis psupsld2.inp.

Substructure load case with nodal pressure loads

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D8 SFM3D4

Features tested

The ability to define a substructure load case with nodal pressure loads and apply it on the usage level is tested.

Problem description

A simple wing model is discretized using C3D8 elements. Kinematic and distributing coupling constraints connect reference nodes on the tip and the top surface of the wing, respectively. A coarse, almost regular, nonstructural load mesh is generated using SFM3D4 elements and tied to the wing structure using tie constraints.

The substructure is defined from the wing structure and the tied load mesh. Both displacement degrees of freedom are retained for the coupling's reference nodes, and load vectors representing nodal pressure loadings are defined on the load mesh using distributed pressure load type NP. The tie constraints impose the mapping of the nodal pressure load distributions on the wing structure. Such nodal pressure load cases might be used, for example, to apply transient distributed loads like lift forces or contact pressure with high accuracy at the usage level. In this analysis, a load case is selected and activated in a static analysis at the usage level. The results recovered from the substructure analysis are compared with the results from the analysis without a substructure.

Results and discussion

The results from the substructure analysis exactly match the results that are obtained when the substructure is not used.

Input files

substr_wing_use.inp	Nodal pressure loading analysis using substructure.
substr_win_gen.inp	Substructure generation file referenced in the analysis `substr_wing_use.inp`.
nosubstr_wing.inp	Nodal pressure loading analysis without substructure.

Thermal-stress analysis with substructures

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

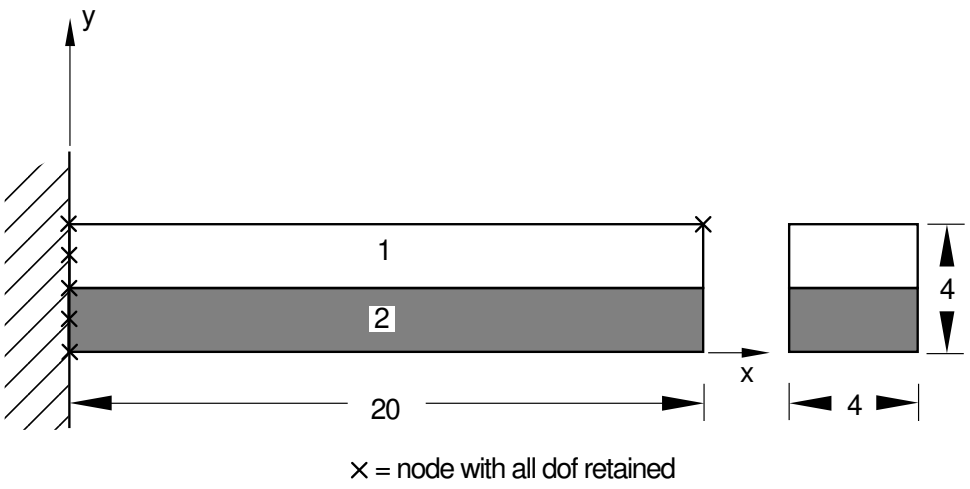
B21 CPS4R

Features tested

The ability to perform thermal stress analysis using substructures is tested.

Problem description

In the first analysis a cantilevered bimetallic beam is discretized using CPS4R elements. Both displacement degrees of freedom are retained for all nodes at the fixed end and for the tip of the beam. A substructure load case is used to define a temperature load over all nodes comprising the substructure, and a uniform increase in temperature is subsequently prescribed on the usage level.



In the second analysis a substructure is generated from a single B21 element and is used to test thermal preloading of substructures. All degrees of freedom are constrained at one end of the beam, whereas the other end is allowed to expand axially. In the preload step the beam is raised to a temperature of 100°. During the analysis the substructure load case is used to apply a temperature of 100° over the entire beam.

In the third analysis a cantilevered bimetallic beam is discretized using CPS4R elements. Both displacement degrees of freedom are retained for all nodes at the fixed end and for the tip of the beam. The substructure load case is used to define a temperature load over all nodes comprising the substructure, and a uniform increase in temperature is prescribed subsequently on the usage level. During substructure generation, controls are set to specify that output of element or nodal information will not be required within the substructure, which reduces the size of the substructure database file.

Results and discussion

The results for the first and third analyses are identical for the analyses performed with and without substructures. The tip deflection of the beam (Node 511) is 2.060 in the vertical direction. In the third analysis the size of the substructure database file is reduced.

The displacements reported on the global level for node 2 in the second analysis are identical to those reported on the substructure level.

Input files

psupthm1.inp	CPS4R elements.
psupthm1_gen.inp	Substructure generation file referenced in the analysis psupthm1.inp.
psupthm2.inp	B21 elements.
psupthm2_gen.inp	Substructure generation file referenced in the analysis psupthm2.inp.
psupthm3.inp	CPS4R elements.
psupthm3_gen.inp	Substructure generation file with **SUBSTRUCTURE GENERATE*, RECOVERY MATRIX=NO, referenced in the analysis psupthm3.inp.

Substructure preload history

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Effects on element and nodal output variables

Elements tested

T2D2

Features tested

Preloading of a substructure followed by perturbation and general steps and the recovery of nodal and element variables.

Problem description

A substructure is formed from a one-element truss model constructed of an elastic-plastic material. The substructure will be subjected to a preload (axial force) that causes inelastic strains. The substructure stiffness matrix is then formed about this base state. Additional loads are applied during global usage through a perturbation step and a general step.

Results and discussion

STEP	LOAD	S11	E11	EE11	PEEQ
Preload	Δu=.0015	32721	1.499E−3	1.0907E−3	4.082E−4
Perturbation	1000	1003	3.342E−5	3.342E−5	n/a
General	1000	1003	3.342E−5	3.342E−5	4.082E−6

The results from the global analysis are consistent with the assumptions of substructures. Namely, the elastic stiffness is used during substructure generation, and initial stress stiffening effects are considered. The stresses and strains from both steps are in addition to the values from the preload step.

Input files

psuppre1.inp	Input file for this analysis.
psuppre1_gen1.inp	Substructure generation referenced in the analysis psuppre1.inp.

Effects of nonlinearities on the stiffness matrix

Elements tested

T2D2

Features tested

Effects of material and geometric nonlinearities on the resulting stiffness matrix of a substructure.

Problem description

Two substructures are created from single-element truss models, one made of a pure elastic material and the other made of an elastic-plastic material. Young's modulus is 3.0E5 in both models, and both structures are subject to a preload (prescribed displacement). The effects of the nonlinearities are incorporated into the static analysis by using the NLGEOM parameter. The magnitude of the applied load is high enough to ensure plastic deformation in the elastic-plastic material. The tangent stiffness value, k_{11}, obtained for each case is compared to the corresponding value obtained by the analysis of an analogous global model without substructures.

Results and discussion

	Substructure		No substructure	
	NLGEOM	NO NLGEOM	NLGEOM	NO NLGEOM
Pure elastic	2.243E5	3.000E5	2.243E5	3.000E5
Elastic-plastic	2.403E5	3.000E5	2.403E5	3.000E5

For the substructure models analyzed without the NLGEOM parameter, the substructure stiffness is the elastic stiffness itself, and material nonlinearities such as plasticity are not accounted for during the creation of the substructure. However, when NLGEOM is used in the preload history definition, the effects of stress stiffening and material nonlinearity are accounted for.

Input files

psuppre2lg_elastic.inp	Substructure without NLGEOM and elastic material properties.
psuppre2lg_elastic_plastic.inp	Substructure without NLGEOM and elastic plastic material properties.
psuppre2nl_elastic.inp	Substructure with NLGEOM and elastic material properties.
psuppre2nl_elastic_plastic.inp	Substructure with NLGEOM and elastic plastic material properties.
psupreg2lg.inp	Regular element without NLGEOM.
psupreg2nl.inp	Regular element with NLGEOM.

Effects of contact constraints on the stiffness matrix

Elements tested

CPE4

Features tested

Effects of contact constraints on the resulting stiffness matrix of a substructure.

Problem description

A substructure is formed from a one-element model constructed of an elastic material. A rigid surface consisting of R2D2 elements is moved down to compress the element in the first step. In the second step the element is moved across the rigid surface to generate frictional forces at the contact interface. The substructure stiffness matrix is then formed about this base state. Additional loads are applied during global usage through a perturbation step.

Results and discussion

The results from the global analysis are consistent with the assumptions of substructures; i.e., the nodes on the secondary surface that are in contact prior to the generation of the substructure stiffness matrix are tied to the main surface. The stresses and strains are in addition to the values obtained from the preload steps.

Input files

psupcontact.inp	Input file for this analysis.
psupcontact_gen.inp	Substructure generation referenced in the analysis psupcontact.inp.

Substructure removal

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

CPE8R

Features tested

Removal of substructures on the global level is tested.

Problem description

A model is constructed of three substructures to represent an excavation process. First, a gravity load is applied to the elements, causing them to deform under their own weight. The top substructure is then removed, which causes the bottom layers to expand because of the relief in load. The middle layer is then removed, causing further expansion of the bottom layer.

Results and discussion

The results from the substructure analysis exactly match the results that are obtained when substructures are not used.

Input files

psuprem1.inp	Input file for this analysis.
psuprem1_top.inp	Top substructure generation file referenced in analysis psuprem1.inp.
psuprem1_middle.inp	Middle substructure generation file referenced in analysis psuprem1.inp.
psuprem1_bottom.inp	Bottom substructure generation file referenced in analysis psuprem1.inp.

Substructure damping

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

T3D2

Features tested

Damping of substructures.

Problem description

The model consists of two substructures, each generated from truss elements of type T3D2. The model is oriented along the *x*-axis and is constrained at one end in all degrees of freedom. It is free to move only in the *x*-direction at its other end. In each case the truss is displaced by 25.4 mm (1 in) at its free end in an initial static step. It is then released in the direct-integration implicit dynamic (or modal dynamic) step, and the displacement response history is saved on a file for postprocessing.

Results and discussion

The results from the substructure analysis exactly match the results that are obtained when substructures are not used.

Input files

psupdmp1.inp	Material damping at substructure generation level.
psupdmp1_gen1.inp	Substructure generation file referenced in the analysis psupdmp1.inp.
psupdmp2.inp	Modal damping at the global level.
psupdmp2_gen1.inp	Substructure generation file referenced in the analysis psupdmp2.inp.
psupdmp3.inp	Material damping at generation level is overridden at the global level.
psupdmp3_gen1.inp	Substructure generation file referenced in the analysis psupdmp3.inp.

Substructures with rebar

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

CAX4 MAX1 MAX2 MGAX1 MGAX2 SAX1 SAX2 SFMAX1 SFMAX2
SFMGAX1 SFMGAX2

Features tested

Rebar in substructures.

Problem description

Generation files *psupreb1_gen1.inp*, *psupreb2_gen1.inp*, *psupreb3_gen1.inp*, and *psupreb4_gen1.inp* are made up of single elements, with each element being treated as a substructure for the purposes of this problem. For the hoop tests a substructure load case is used to define a pressure load acting on the surface of the shell, and a uniform pressure is subsequently applied at the usage level. For the radial rebar tests a substructure load case is used to define a concentrated load, and it is applied at the usage level at the outer edge of the shell.

Generation files *psupreb5_gen1.inp*, *psupreb6_gen1.inp*, *psupreb7_gen1.inp*, *psupreb8_gen1.inp*, *psupreb10_gen1.inp*, *psupreb11_gen1.inp*, *psupreb12_gen1.inp*, and *psupreb13_gen1.inp* are made up of five elements each, all five elements being combined to form a single substructure. In input files *psupreb5.inp*, *psupreb6.inp*, *psupreb10.inp*, and *psupreb11.inp* the substructure model is subjected to axial tension. In input files *psupreb7.inp*, *psupreb8.inp*, *psupreb12.inp*, and *psupreb13.inp* a two-step loading is applied, with axial tension applied in the first step and a torsional moment in the second. Both loadings are applied by using the substructure loading.

Generation files *psupreb9_gen1.inp* and *psupreba_gen1.inp* are made up of five elements each, all five elements being combined to form a single substructure. The substructure model is subjected to an axial tension applied.

Results and discussion

The results from the substructure analysis exactly match the results that are obtained when substructures are not used.

Input files

psupreb1.inp	SAX1 elements with hoop rebar.
psupreb1_gen1.inp	Substructure generation file referenced in the analysis psupreb1.inp.
psupreb2.inp	SAX1 elements with radial rebar.
psupreb2_gen1.inp	Substructure generation file referenced in the analysis psupreb2.inp.
psupreb3.inp	SAX2 elements with hoop rebar.
psupreb3_gen1.inp	Substructure generation file referenced in the analysis psupreb3.inp.
psupreb4.inp	SAX2 elements with radial rebar.

psupreb4_gen1.inp	Substructure generation file referenced in the analysis psupreb4.inp.
psupreb5.inp	MAX1 elements with rebar under axial tension.
psupreb5_gen1.inp	Substructure generation file referenced in the analysis psupreb5.inp.
psupreb6.inp	MAX2 elements with rebar under axial tension.
psupreb6_gen1.inp	Substructure generation file referenced in the analysis psupreb6.inp.
psupreb7.inp	MGAX1 elements with rebar under axial tension and twist.
psupreb7_gen1.inp	Substructure generation file referenced in the analysis psupreb7.inp.
psupreb8.inp	MGAX2 elements with rebar under axial tension and twist.
psupreb8_gen1.inp	Substructure generation file referenced in the analysis psupreb8.inp.
psupreb9.inp	CAX4 elements with hoop rebar.
psupreb9_gen1.inp	Substructure generation file referenced in the analysis psupreb9.inp.
psupreb10.inp	SFMAX1 elements with rebar under axial tension.
psupreb10_gen1.inp	Substructure generation file referenced in the analysis psupreb10.inp.
psupreb11.inp	SFMAX2 elements with rebar under axial tension.
psupreb11_gen1.inp	Substructure generation file referenced in the analysis psupreb11.inp.
psupreb12.inp	SFMGAX1 elements with rebar under axial tension and twist.
psupreb12_gen1.inp	Substructure generation file referenced in the analysis psupreb12.inp.
psupreb13.inp	SFMGAX2 elements with rebar under axial tension and twist.
psupreb13_gen1.inp	Substructure generation file referenced in the analysis psupreb13.inp.
psupreba.inp	CAX4 elements with axial rebar.
psupreba_gen1.inp	Substructure generation file referenced in the analysis psupreba.inp.

Frequency extraction for substructures

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

B21

Features tested

Frequency extraction analysis for substructures.

Problem description

The substructures defined in each generation file are identical, each consisting of 10 B21 beam elements. In each case one of the substructure's ends is fixed. In addition, in the file *psupfre2_gen1.inp* a local coordinate system is defined for all nodes, and in the file *psupfre3_gen1.inp* the substructure is rotated by 90° using the substructure property definition.

Results and discussion

The results from the substructure analysis match the results that are obtained when the substructures are not used.

Input files

psupfre1.inp	B21 elements.
psupfre1_gen1.inp	Substructure generation file referenced in the analysis psupfre1.inp.
psupfre2.inp	B21 elements; a local coordinate system is defined for all nodes using the **TRANSFORM* option.
psupfre2_gen1.inp	Substructure generation file referenced in the analysis psupfre2.inp.
psupfre3.inp	B21 elements; the substructure is rotated using the **SUBSTRUCTURE PROPERTY* option.
psupfre3_gen1.inp	Substructure generation file referenced in the analysis psupfre3.inp.

Substructures with large rotations

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Substructures moving as rigid bodies

Features tested

Substructure's ability to move as a rigid body. The substructures undergo large rotation motions in analyses that generate negligible strain/stress in the substructure. Both static and direct-integration implicit dynamic analyses are verified.

Problem description

A rectangular substructure is formed. The substructure is subjected to boundary conditions and concentrated loads specified at the retained degrees of freedom that create negligible strain in the substructure but generate large rotations of the model. In the static analyses the substructure is constrained using springs to prevent numerical singularities.

A second identical mesh is defined without substructures. The displacements, rotations, and reaction forces should be nearly identical between the two equivalent analyses.

Results and discussion

All results in the substructure are nearly identical to the results in the regular mesh.

Input files

substr_rbm_solid2d_sta.inp	Large rotations, static, two-dimensional analysis using substructures.
substr_rbm_solid2d_dyn.inp	Large rotations, direct-integration implicit dynamic, two-dimensional analysis using substructures.
substr_rbm_solid2d_gen.inp	Substructure generation file referenced in the two files above.
nosubstr_rbm_solid2d_sta.inp	Large rotations, static, two-dimensional analysis without substructures.
nosubstr_rbm_solid2d_dyn.inp	Large rotations, direct-integration implicit dynamic, two-dimensional analysis without substructures.
substr_rbm_solid3d_sta.inp	Large rotations, static, three-dimensional analysis using substructures.
substr_rbm_solid3d_dyn.inp	Large rotations, direct-integration implicit dynamic, three-dimensional analysis using substructures.
substr_rbm_solid3d_gen.inp	Substructure generation file referenced in the two files above.
nosubstr_rbm_solid3d_sta.inp	Large rotations, static, three-dimensional analysis without substructures.
nosubstr_rbm_solid3d_dyn.inp	Large rotations, direct-integration implicit dynamic, three-dimensional analysis without substructures.
substr_rbm_beam2d_sta.inp	Large rotations, static, two-dimensional analysis using substructures.

substr_rbm_beam2d_dyn.inp	Large rotations, direct-integration implicit dynamic, two-dimensional analysis using substructures.
substr_rbm_beam2d_gen.inp	Substructure generation file referenced in the two files above.
nosubstr_rbm_beam2d_sta.inp	Large rotations, static, two-dimensional analysis without substructures.
nosubstr_rbm_beam2d_dyn.inp	Large rotations, direct-integration implicit dynamic, two-dimensional analysis without substructures.
substr_rbm_beam3d_sta.inp	Large rotations, static, three-dimensional analysis using substructures.
substr_rbm_beam3d_dyn.inp	Large rotations, direct-integration implicit dynamic, three-dimensional analysis using substructures.
substr_rbm_beam3d_gen.inp	Substructure generation file referenced in the two files above.
nosubstr_rbm_beam3d_sta.inp	Large rotations, static, three-dimensional analysis without substructures.
nosubstr_rbm_beam3d_dyn.inp	Large rotations, direct-integration implicit dynamic, three-dimensional analysis without substructures.

Small-deformation substructures in large rotations

Features tested

Substructures that are subject to elastic small-deformations but undergo large rotations. Both static and direct-integration implicit dynamic analyses are verified.

Problem description

The modes to be used are specified, and a rectangular mesh is formed. The loading and boundary contions specified at the retained degrees of freedom are such that elastic small-strain-inducing defomations occur on top of large rotations of the substructure. In the static analyses additional springs are used to prevent numerical singularities. Results are then compared to results obtained from equivalent analyses that do not use substructures.

Results and discussion

All results in the analyses using substructures are nearly identical to the results obtained in the analyses using a regular mesh.

Input files

substr_smdef_solid2d_sta.inp	Elastic, small-strain, large rotations, static, two-dimensional analysis using substructures.
substr_smdef_solid2d_dyn.inp	Elastic, small-strain, large rotations, direct-integration implicit dynamic, two-dimensional analysis using substructures.
substr_smdef_solid2d_gen.inp	Substructure generation file referenced in the two files above.
nosubstr_smdef_solid2d_sta.inp	Elastic, small-strain, large rotations, static, two-dimensional analysis without substructures.
nosubstr_smdef_solid2d_dyn.inp	Elastic, small-strain, large rotations, direct-integration implicit dynamic, two-dimensional analysis without substructures.
substr_smdef_solid3d_sta.inp	Elastic, small-strain, large rotations, static, three-dimensional analysis using substructures.
substr_smdef_solid3d_dyn.inp	Elastic, small-strain, large rotations, direct-integration implicit dynamic, three-dimensional analysis using substructures.
substr_smdef_solid3d_gen.inp	Substructure generation file referenced in the two files above.

nosubstr_smdef_solid3d_sta.inp	Elastic, small-strain, large rotations, static, three-dimensional analysis without substructures.
nosubstr_smdef_solid3d_dyn.inp	Elastic small-strain large rotations direct-integration implicit dynamic three-dimensional analysis without substructures.
substr_smdef_beam2d_sta.inp	Elastic, small-strain, large rotations, static, two-dimensional analysis using substructures.
substr_smdef_beam2d_dyn.inp	Elastic, small-strain, large rotations, direct-integration implicit dynamic, two-dimensional analysis using substructures.
substr_smdef_beam2d_gen.inp	Substructure generation file referenced in the two files above.
nosubstr_smdef_beam2d_sta.inp	Elastic, small-strain, large rotations, static, two-dimensional analysis without substructures.
nosubstr_smdef_beam2d_dyn.inp	Elastic, small-strain, large rotations, direct-integration implicit dynamic, two-dimensional analysis without substructures.
substr_smdef_shell3d_sta.inp	Elastic, small-strain, large rotations, static, three-dimensional analysis using substructures.
substr_smdef_shell3d_dyn.inp	Elastic, small-strain, large rotations, direct-integration implicit dynamic, three-dimensional analysis using substructures.
substr_smdef_shell3d_gen.inp	Substructure generation file referenced in the two files above.
nosubstr_smdef_shell3d_sta.inp	Elastic, small-strain, large rotations, static, three-dimensional analysis without substructures.
nosubstr_smdef_shell3d_dyn.inp	Elastic, small-strain, large rotations, direct-integration implicit dynamic, three-dimensional analysis without substructures.

User-specified rotation/mirroring and transformations

Features tested

User-rotated or mirrored substructures that also exhibit elastic small-strain deformation in addition to large rotations.

Problem description

A rectangular mesh is formed. At the usage level the substructure is either translated and rotated or mirrored.

A second identical mesh is defined without using substructures but accounting for the user-specified rotation/mirroring. The displacements, rotations, and stresses should be nearly identical between the two equivalent analyses.

Results and discussion

All results in the substructure are nearly identical to the results in the regular mesh.

Input files

substr_urot_shell3d_sta.inp	User-rotated substructure with large rotation motions, static, three-dimensional analysis.
nosubstr_urot_shell3d_sta.inp	Equivalent regular mesh static three-dimensional analysis.
substr_umir_shell3d_sta.inp	User-mirrored substructure with large rotation motions, static, three-dimensional analysis.
nosubstr_umir_shell3d_sta.inp	Equivalent regular mesh static three-dimensional analysis.

substr_user_shell3d_gen.inp	Substructure generation file referenced in files substr_urot_shell3d_sta.inp and substr_umir_shell3d_sta.inp.

Multi-level substructures in large rotations

Features tested

Multi-level substructures that undergo large rotations.

Problem description

Three levels of substructures are created for this particular analysis. The lowest level is a 2×2 mesh of CPE4 elements. The next level comprises two of the first-level substructures, and the third level is the actual structure. The use of unsorted retained degrees of freedom is tested during the creation levels. The loading and boundary conditions specified at the retained degrees of freedom are such that elastic small-strain-inducing defomations occur in addition to the large rotations of the substructure. A second identical mesh is defined without substructures and the results are compared.

Results and discussion

All results in the substructure are nearly identical to the results in the regular mesh.

Input files

substr_multi_solid2d_gen1.inp	Lowest level substructure generation file.
substr_multi_solid2d_gen2.inp	Second level substructure generation file.
substr_multi_solid2d_gen3.inp	Third level substructure generation file.
substr_multi_solid2d_sta.inp	Large rotations, static, two-dimensional analysis.
nosubstr_multi_solid2d_sta.inp	Large rotations, static, two-dimensional analysis without substructures.

Substructures and gravity loads

Features tested

Substructures subjected to fixed direction gravity loads.

Problem description

A rectangular substructure is formed. A gravity load is then applied by accounting for the substructure's gravity load vectors during generation and by specifying a distributed load type GRAV at the substructure usage level. The loading is such that the substructure undergoes large rotations. An equivalent regular mesh is also created, and the results are compared.

Results and discussion

All results in the substructure are nearly identical to the results in the regular mesh.

Input files

substr_grav_solid2d_sta.inp	Large rotations, gravity-loaded, static, two-dimensional analysis using substructures.

substr_grav_solid2d_dyn.inp	Large rotations, gravity-loaded, direct-integration implicit dynamic, two-dimensional analysis using substructures.
substr_grav_solid2d_gen.inp	Substructure generation file referenced in the two files above.
nosubstr_grav_solid2d_sta.inp	Large rotations, gravity-loaded, static, two-dimensional analysis without substructures.
nosubstr_grav_solid2d_dyn.inp	Large rotations, gravity-loaded, direct-integration implicit dynamic, two-dimensional analysis without substructures.
substr_grav_shell3d_sta.inp	Large rotations, gravity-loaded, static, three-dimensional analysis using substructures.
substr_grav_shell3d_dyn.inp	Large rotations, gravity-loaded, direct-integration implicit dynamic, three-dimensional analysis using substructures.
substr_grav_shell3d_gen.inp	Substructure generation file referenced in the two files above.
nosubstr_grav_shell3d_sta.inp	Large rotations, gravity-loaded, static, three-dimensional analysis without substructures.
nosubstr_grav_shell3d_dyn.inp	Large rotations, gravity-loaded, direct-integration implicit dynamic, three-dimensional analysis without substructures.

Substructures, connector elements, coupling constraints, and rigid body constraints

Features tested

Multiple substructures connected with connector elements and coupling constraints in large motions. Substructures included in a rigid body constraint in large rotations. How to switch quickly from a rigid body model of a part to a small-strain large-motion representation of the same part.

Problem description

The common 4-bar mechanism is analyzed (see *Overconstraint Checks*). The two-dimensional rigid bodies are meshed using CPE4 elements. The coupling constraints are used to attach connection nodes to the ends of each bar, and connector elements are used to enforce the appropriate kinematic constraints between the bars. The bars are gravity loaded, and the mechanism is driven by prescribing the available components of relative motion in connector elements. Since the four bars are identical in shape, only one substructure is generated. The substructure is then translated, mirrored, and rotated at the usage level to create four copies of the substructure in the appropriate locations. Results from both static and direct-integration implicit dynamic analyses are verified against equivalent analyses that do not use substructures.

In addition, at the usage level one of the substructures is turned into a rigid part. The attached input files illustrate how one can very efficiently switch from a rigid (faster to run) model (*substr_4barrb_solid2d_sta.inp* and *nosubstr_4barrb_solid2d_sta.inp*) to a small-deformation large-rotations efficient subtructure representation of the same model (*substr_4bar_solid2d_sta.inp*). The substructure analysis is typically significantly faster to run than the regular mesh models (*nosubstr_4bar_solid2d_sta.inp*).

Results and discussion

All results in the substructure are nearly identical to the results in the regular mesh.

Input files

substr_4bar_solid2d_gen.inp	Substructure generation file for one bar in the mechanism.
substr_4bar_solid2d_sta.inp	Static analysis of the gravity-loaded 4-bar mechanism using substructures.

substr_4barrb_solid2d_sta.inp	Direct-integration implicit dynamic analysis of the 4-bar multibody with one substructure included in a rigid body definition.
nosubstr_4bar_solid2d_sta.inp	Static analysis of the gravity-loaded 4-bar mechanism without substructures.
nosubstr_4barrb_solid2d_sta.inp	Direct-integration implicit dynamic analysis of the 4-bar multibody using four separate rigid bodies and no substructures.
substr_4bar_solid2d_dyn.inp	Direct-integration implicit dynamic analysis of the gravity-loaded 4-bar multibody using substructures.
nosubstr_4bar_solid2d_dyn.inp	Direct-integration implicit dynamic analysis of the gravity-loaded 4-bar multibody without substructures.

Substructures and contact

Features tested

Large rotation substructures and contact.

Problem description

A rectangular substructure is formed. The applied loads and boundary conditions are such that the substructure exhibits large rotations. After a 45° rotation, impact with a rigid surface occurs. Results are compared with results from an equivalent model without substructures.

Results and discussion

All results in the substructure are nearly identical to the results in the regular mesh.

Input files

substr_contact_solid3d_dyn.inp	Large rotations, direct-integration implicit dynamic, three-dimensional analysis using substructures and contact.
substr_contact_solid3d_gen.inp	Substructure generation reference in the file above.
nosubstr_contact_solid3d_dyn.inp	Large rotations, direct-integration implicit dynamic, three-dimensional analysis without substructures and contact.

Miscellaneous tests

Features tested

Substructures with large rotations with multi-point constraints, model change, prescribed initial conditions, and restart analysis.

Problem description

Several input files are created to test various features with large rotation substructures. Results are compared with equivalent models that do not use substructures.

Results and discussion

All results in the substructure are nearly identical to the results in the regular mesh.

Input files

substr_misc_solid2d_gen.inp	Substructure generation referenced in the files below.
substr_misc_solid2d_dyn.inp	Direct-integration implicit dynamic, two-dimensional analysis using substructures, multi-point constraints, and initial conditions.
nosubstr_misc_solid2d_dyn.inp	Direct-integration implicit dynamic, two-dimensional analysis without substructures and **MPC*s and initial conditions.
substr_misc_solid2d_res.inp	Restart analysis from substr_misc_solid2d_dyn.inp using substructure.
nosubstr_misc_solid2d_res.inp	Restart analysis from nosubstr_misc_solid2d_dyn.inp without substructures.
substr_misc_solid2d_modelch.inp	Model change analysis using substructures.
nosubstr_misc_solid2d_modelch.inp	Model change analysis without substructures.
substr_misc_beam_1node_gen.inp	Substructure generation referenced in the file below.
substr_misc_beam_1node.inp	One retained node analysis using substructures.
nosubstr_misc_beam_1node.inp	Equivalent analysis without substructures.
substr_misc_pert_gen.inp	Substructure generation referenced in the files below.
substr_misc_nlgrot_pert.inp	Perturbation step after a geometrically nonlinear static rotation.
substr_misc_urot_pert.inp	Perturbation step after a user-specified rotation.

Coupled structural-acoustic analysis with substructures

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Features tested

Generation and usage of coupled structural-acoustic substructures.

Basic coupled structural-acoustic substructure analysis

Features tested

The ability to generate a substructure from a simple coupled structural-acoustic model. The substructure is then used in various analysis types.

Problem description

A very simple three-element mesh is used. Two solid (CPE4) elements are tied to a single acoustic (AC2D4) element. The substructure is generated using all eigenmodes that can be extracted, and a substructure load case is generated for a concentrated load. The substructure is then used in a frequency extraction analysis followed by several dynamic procedures (steady-state dynamic response, modal dynamic analysis, and direct-integration implicit dynamic analysis).

A second identical mesh is defined without substructures. The results recovered from the substructure analysis are then compared with the results from the analysis without substructures.

Results and discussion

The results recovered from the substructure analysis are identical to the results from the analysis without substructures.

Input files

substr_small_ac2d4_gen.inp	Coupled structural-acoustic substructure generation input file.
substr_small_ac2d4_use.inp	Coupled structural-acoustic substructures in several dynamic procedures.
nosubstr_small_ac2d4.inp	Input file for the equivalent analysis without substructures.

Coupled structural-acoustic substructure analysis of a box filled with fluid

Features tested

Coupled structural-acoustic substructure generation and usage of a more complex model.

Problem description

The mesh is the same as the box model described in *Adaptive meshing applied to coupled structural-acoustic problems*. The box is preloaded by a surface-distributed load applied to the interior of the box in a static step. The substructure is generated using a large number of eigenmodes and then used in a frequency extraction analysis followed by several dynamic procedures (steady-state dynamic response, modal dynamic analysis, and direct-integration implicit dynamics).

A second identical mesh is defined without substructures. The results recovered from the substructure analysis are compared with the results from the analysis without substructures.

Results and discussion

The results recovered from the substructure analysis compare very well with the results from the analysis without substructures.

Input files

substr_box_ac2d4_gen.inp	Two-dimensional coupled structural-acoustic substructure generation input file.
substr_box_ac2d4_use.inp	Two-dimensional coupled structural-acoustic substructure used in several steady-state dynamic analysis steps.
nosubstr_box_ac2d4.inp	Input file for the equivalent analysis without substructures.
substr_box_ac2d4_moddyn_gen.inp	Two-dimensional coupled structural-acoustic substructure generation input file.
substr_box_ac2d4_moddyn_use.inp	Two-dimensional coupled structural-acoustic substructure used in direct-integration implicit dynamic and modal dynamic analysis steps.
nosubstr_box_ac2d4_moddyn.inp	Input file for the equivalent analysis without substructures.
substr_box_ac3d4_gen.inp	Three-dimensional coupled structural-acoustic substructure generation input file.
substr_box_ac3d4_use.inp	Three-dimensional coupled structural-acoustic substructure used in several steady-state dynamic response steps.
nosubstr_box_ac3d4.inp	Input file for the equivalent analysis without substructures.
substr_box_ac3d4_moddyn_gen.inp	Three-dimensional coupled structural-acoustic substructure generation input file.
substr_box_ac3d4_moddyn_use.inp	Three-dimensional coupled structural-acoustic substructure used in direct-integration implicit dynamic and modal dynamic analysis steps.
nosubstr_box_ac3d4_moddyn.inp	Input file for the equivalent analysis without substructures.

Coupled structural-acoustic substructure analysis of a simple axisymmetric tire model

Features tested

Coupled structural-acoustic substructure generation and usage.

Problem description

The mesh is the same as the axisymmetric tire model described in *Adaptive meshing applied to coupled structural-acoustic problems*. The substructure is generated using a large number of retained eigenmodes, and a substructure load case is generated for a concentrated load. The substructure is then used in a frequency

extraction analysis followed by several dynamic procedures (steady-state dynamic response, modal dynamic analysis, and direct-integration implicit dynamic analysis).

A second identical mesh is defined without substructures. The results recovered from the substructure analysis are then compared with the results from the analysis without substructures.

Results and discussion

The results recovered from the substructure analysis compare very well with the results from the analysis without substructures.

Input files

substr_smalltire_acax4_gen.inp	Axisymmetric coupled structural-acoustic substructure generation input file.
substr_smalltire_acax4_use.inp	Axisymmetric coupled structural-acoustic substructures used in several dynamic procedures.
nosubstr_smalltire_acax4.inp	Input file for the equivalent analysis without substructures.

Additive manufacturing process simulation

In this section:

- *Heat energy balance*
- *Pattern-based eigenstrain analysis*
- *Trajectory-based eigenstrain analysis*
- *Metallurgical phase transformations*
- *Material deposition*
- *Moving heat sources*
- *Material removal*

Heat energy balance

Products: Abaqus/Standard

References:

- *Thermomechanical Simulation of Additive Manufacturing Processes*
- *Special-Purpose Techniques for Additive Manufacturing*

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

DC3D4 DC3D8

Features tested

This section provides basic verification tests of energy balance in heat transfer analyses of powder bed-type additive manufacturing processes where:

- the material deposition is simulated by progressive element activation; and
- the heating induced by a moving heat source is simulated by a moving heat flux.

Problem description

Each model simulates powder bed-type additive manufacturing processes of three disconnected cubes with dimensions 1.0 × 1.0 × 1.0. The analyses use the special-purpose techniques for powder bed-type additive manufacturing processes in Abaqus/Standard (see *Thermomechanical Analysis of Powder Bed–Type Additive Manufacturing Processes Using the Trajectory-Based Method* and *Thermomechanical Analysis of Powder Bed–Type Additive Manufacturing Processes Using the Pattern-Based Method*).

Model Different types of progressive element activation are applied to the cubes to simulate the layer-upon-layer material deposition:

Cube 1	Progressively activated by partial element activation
Cube 2	Progressively activated by full element activation (that is, elements can be only inactive or fully activated)
Cube 3 (reference)	No element activation (that is, elements are always active)

The material deposition motion of a cube is shown in *Figure 1*. The line in the figure represents the path traveled by the recoater (or roller blade) as new layers of powder material are deposited. The material layer thickness is 0.05. A total of 20 layers is required to build one cube. An event series that represents the path of the recoater is used in the trajectory-based method. For the pattern-based method, the layer thickness is defined in a parameter table.

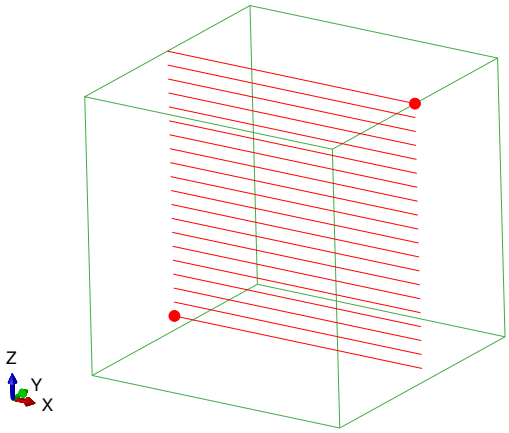


Figure 1: The recoater or roller blade motion.

Mesh

Each cube is modeled by a mesh of 29 DC3D4 elements or a mesh of 3 × 3 × 3 DC3D8 elements.

Material

All cubes have the same constant thermal properties.

Conductivity	$k = 5000$
Density	$\rho = 8000$
Specific heat	$c_p = 500$

(The units are not important.)

Loading

A moving heat flux is applied to each cube to simulate the progressive heating by a moving heat source (such as a laser). As shown in *Figure 2*, the scanning path of the heat source for each layer consists of a 74° hatching pattern with 0.075 spacing followed by a contour. There is no rotation between each layer.

The scanning is performed with a laser power $P = 1 \times 10^9$, with the energy absorption efficiency as $\eta = 100\%$. The laser beam spot size is assumed to have a radius of 0.025 at the intersection with the part surface. The penetration depth is assumed to be 0.025. Three types of energy distribution of the laser are tested: concentrated point source, uniform distribution in a finite box, and Goldak double ellipsoid distribution. An event series that represents the path of the heat source is used in the trajectory-based method. For the pattern-based method, the path of the heat source is not used explicitly. Abaqus estimates the amount of energy deposited into the structure based on the scan pattern definition. The difference between the total heat input into a layer estimated from the scan pattern definition and that estimated from the trajectory is not expected to be significant. The energy distribution is assumed to be a concentrated point source when the pattern-based method is used.

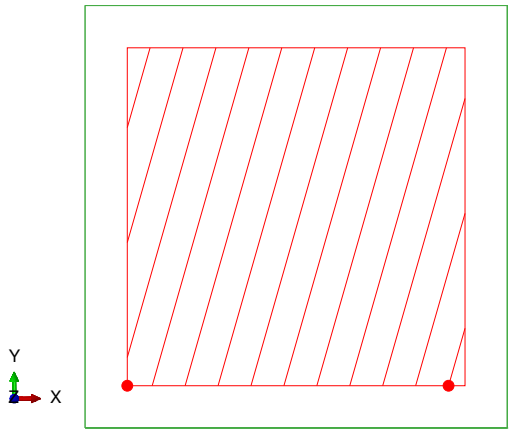


Figure 2: The laser scanning path on each layer.

Boundary conditions and initial conditions

Each cube is insulated on all edges. The initial temperature of newly deposited material is $\theta_i = 0.0$.

Results and discussion

Based on the heat energy balance, energy input by the laser should be equal to the increase of the internal heat energy of the cube. Therefore, the final steady-state temperature of a cube can be computed analytically,

$$\theta_f = \theta_i + \frac{\eta P \Delta t}{\rho c_p V} = 39.115,$$

where $\Delta t = 0.15646$ is the total time of the laser scanning and $V = 1.0$ is the volume of the cube.

The simulation results of the final steady-state temperature were found to be within 1% of the analytical solution, as shown in *Table 1* and *Table 2*.

For the trajectory-based method, output variable ALLHUMDFLUX, which is the total heat energy input into the whole model (consisting of three cubes), also agrees with the analytical solution $Q = 3\eta P \Delta t = 4.6938 \times 10^8$.

Abaqus estimates the amount of energy deposited into the structure based on the scan pattern definition when the pattern-based method is used. The output variable ALLHUMDFLUX for one cube indicates the total heat energy input $E = 1.71 \times 10^8$, which is 9.3% higher than the heat energy input in analyses performed using the trajectory-based method. Therefore, the final steady-state temperature of a cube can be computed analytically,

$$\theta_f = \theta_i + \frac{E}{\rho c_p V} = 42.75.$$

Table 1: The final steady-state temperature of analyses using DC3D4 elements and the trajectory-based method.

Heat source energy distribution	Concentrated		Uniform		Goldak	
	Temperature	Error	Temperature	Error	Temperature	Error
Partial element activation	39.116	0.00%	39.099	0.04%	39.433	0.81%

Heat source energy distribution	Concentrated		Uniform		Goldak	
	Temperature	Error	Temperature	Error	Temperature	Error
Full element activation	39.116	0.00%	39.099	0.04%	39.433	0.81%
No element activation (reference)	39.116	0.00%	39.099	0.04%	39.434	0.82%

Table 2: The final steady-state temperature of analyses using DC3D8 elements and the trajectory-based method.

Heat source energy distribution	Concentrated		Uniform		Goldak	
	Temperature	Error	Temperature	Error	Temperature	Error
Partial element activation	39.115	0.00%	39.115	0.00%	39.435	0.82%
Full element activation	39.115	0.00%	39.115	0.00%	39.435	0.82%
No element activation (reference)	39.115	0.00%	39.115	0.00%	39.435	0.82%

Table 3: The final steady-state temperature of analyses using DC3D4 elements and the pattern-based method.

Heat source energy distribution	Temperature	Error
Partial element activation	42.75	0.00%
Full element activation	42.75	0.00%
No element activation (reference)	42.75	0.00%

Table 4: The final steady-state temperature of analyses using DC3D8 elements and the pattern-based method.

Heat source energy distribution	Temperature	Error
Partial element activation	42.75	0.00%
Full element activation	42.75	0.00%
No element activation (reference)	42.75	0.00%

Input files

The trajectory-based method

am_heatbalance_dc3d4_concentrated.inp	DC3D4 elements, concentrated heat source.
am_heatbalance_dc3d4_uniform.inp	DC3D4 elements, heat source.
am_heatbalance_dc3d4_goldak.inp	DC3D4 elements, heat source using the Goldak distribution.
am_heatbalance_dc3d8_concentrated.inp	DC3D8 elements, concentrated heat source.
am_heatbalance_dc3d8_uniform.inp	DC3D8 elements, heat source using a uniform distribution in a finite box.
am_heatbalance_dc3d8_goldak.inp	DC3D8 elements, heat source using the Goldak distribution.

The pattern-based method

am_heatbalance_dc3d4_patternbased_full.inp	DC3D4 elements, full element activation.
am_heatbalance_dc3d4_patternbased_partial.inp	DC3D4 elements, partial element activation.
am_heatbalance_dc3d4_patternbased_iniact.inp	DC3D4 elements, elements are always active.
am_heatbalance_dc3d8_patternbased_full.inp	DC3D8 elements, full element activation.
am_heatbalance_dc3d8_patternbased_partial.inp	DC3D8 elements, partial element activation.
am_heatbalance_dc3d8_patternbased_iniact.inp	DC3D8 elements, elements are always active.

Input files containing definitions or data to be included in the input files listed above

ABQ_am_special_purpose_types.inp	Types of property tables, parameter tables, and event series used by the special-purpose techniques for the simulation of common additive manufacturing processes in Abaqus.
es_heatbalance_laser.inp	Event series data of the laser scanning path used in the trajectory-based method.
es_heatbalance_roller.inp	Event series data of the roller blade motion used in the trajectory-based method.

Pattern-based eigenstrain analysis

Products: Abaqus/Standard

References:

- *Eigenstrain-Based Simulation of Additive Manufacturing Processes*
- *Special-Purpose Techniques for Additive Manufacturing*

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D8 S4

Features tested

This section provides basic verification tests of the pattern-based eigenstrain method used in additive manufacturing processes where:

- the material deposition is simulated by progressively activating elements layer by layer; and
- the elastic deformation of a printing part is driven by eigenstrain loadings.

Problem description

The problem simulates the process of additive manufacturing of a cube of dimensions 20.0 × 20.0 × 20.0. The analyses use the special-purpose technique for pattern-based eigenstrain modeling of additive manufacturing processes in Abaqus/Standard (see *Specifying Element Activation and Eigenstrain Using the Pattern-Based Method*). Models using solid elements (C3D8) and shell elements (S4) are tested.

Model

Full element activation is applied to the cube to simulate the layer-upon-layer material deposition. The material layer thickness (slice height) is 1.0. A total of 20 layers is needed to build one cube.

Mesh

The cube is modeled with a mesh of 20 × 20 × 20 C3D8 elements or a mesh of 16800 S4 elements.

Material

A linear isotropic elastic material is used, with:

Young's modulus,	E=10000
Poisson's ratio,	υ=0.33

(The units are not important.)

Loading

Different scan-pattern strategies are applied to build the lower half and the upper half of the cube, as shown in *Figure 1* and *Figure 2*, respectively. Each scan pattern consists of four patches. The in-plane dimensions of all patches are 5.0 × 5.0. For the lower half of the cube the layer-to-layer (slice-to-slice) rotation of the scan pattern is defined as 0° about the *K*-axis of the build system *I-J-K*; namely, the first 10 layers have an identical eigenstrain pattern. The layer-to-layer rotation angle of the scan pattern of the upper half of the cube is 90°, which means that the eigenstrain pattern repeats every four layers.

The orientation of the build system *I-J-K* is defined to align with the global orientation *X-Y-Z*. The origin of the build system is at the bottom center of the cube.

Orthotropic eigenstrains are applied on element activation, $\varepsilon^*_{12} = \varepsilon^*_{13} = \varepsilon^*_{23} = 0$. *Table 1* shows the value and the in-plane rotation angle of the eigenstrains for each patch. For solid elements (for example, C3D8 elements), the material orientation is updated to align with the rotation angle of the eigenstrain of the patch.

Table 1: Rotation angle and value of eigenstrains of each patch.

Patch No.	Local in-plane rotation	Eigenstrain component ε^*_{11}	Eigenstrain component ε^*_{22}	Eigenstrain component ε^*_{33}
1	0°	-0.001	-0.001	-0.0005
2	0°	-0.002	-0.002	-0.0005
3	90°	-0.003	-0.003	-0.0005
4	90°	-0.004	-0.004	-0.0005
5	45°	-0.005	-0.005	-0.0005
6	45°	-0.006	-0.006	-0.0005
7	-45°	-0.007	-0.007	-0.0005
8	-45°	-0.008	-0.008	-0.0005

Boundary conditions All degrees of freedom of the nodes on the bottom surface of the cube are fixed.

Results and discussion

Both tests using C3D8 and S4 elements have the expected results, as defined in the problem description. The results of eigenstrain component EEIG11 are shown in *Figure 3* and *Figure 4*.

Input files

am_eigenstrain_pattern_c3d8.inp	The model using C3D8 elements.
am_eigenstrain_pattern_s4.inp	The model using S4 elements.

Input files containing definitions or data to be included in the input files listed above

ABQ_am_special_purpose_types.inp	Types of property tables, parameter tables, and event series used by the special-purpose techniques for the simulation of common additive manufacturing processes in Abaqus.

Figures

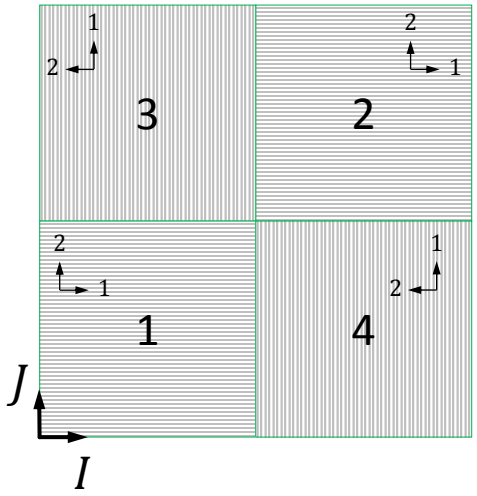


Figure 1: Eigenstrain pattern for building the lower half of the cube.

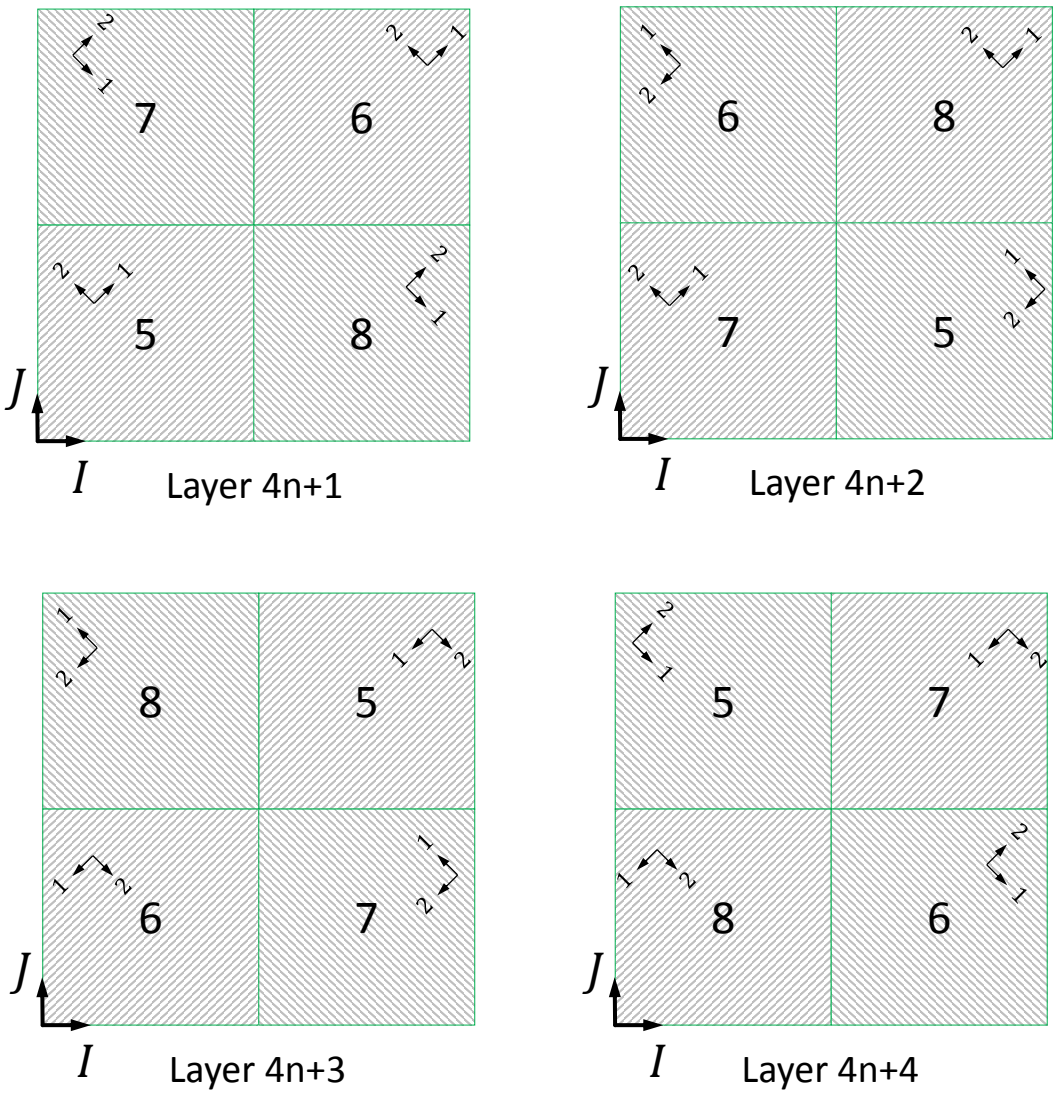


Figure 2: Eigenstrain pattern for building the upper half of the cube.

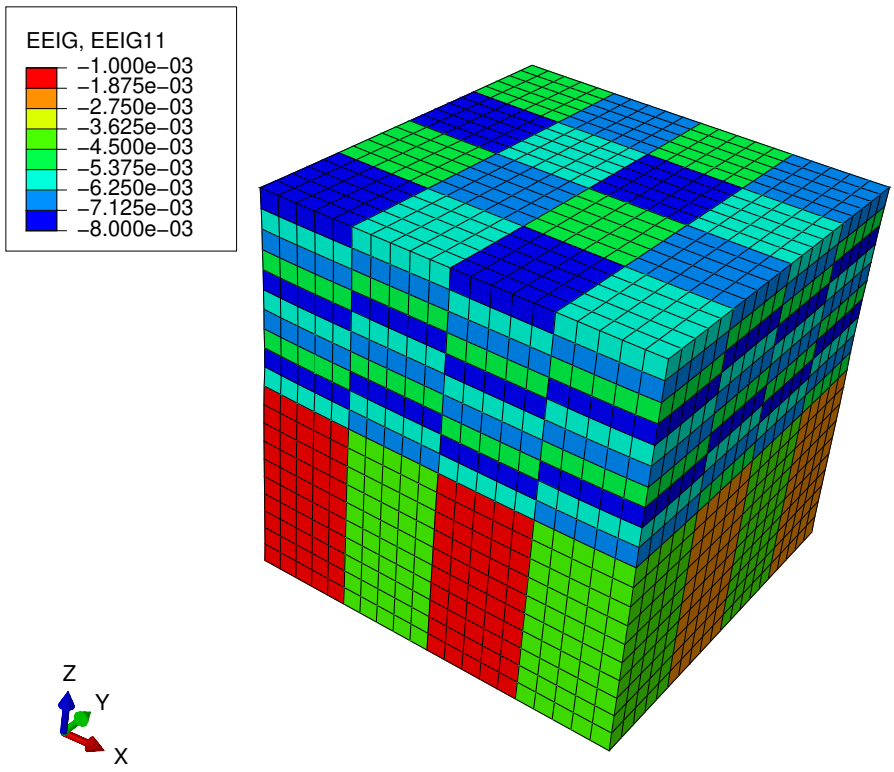


Figure 3: Quilt-style contour plots of eigenstrain component EEIG11 (C3D8 elements).

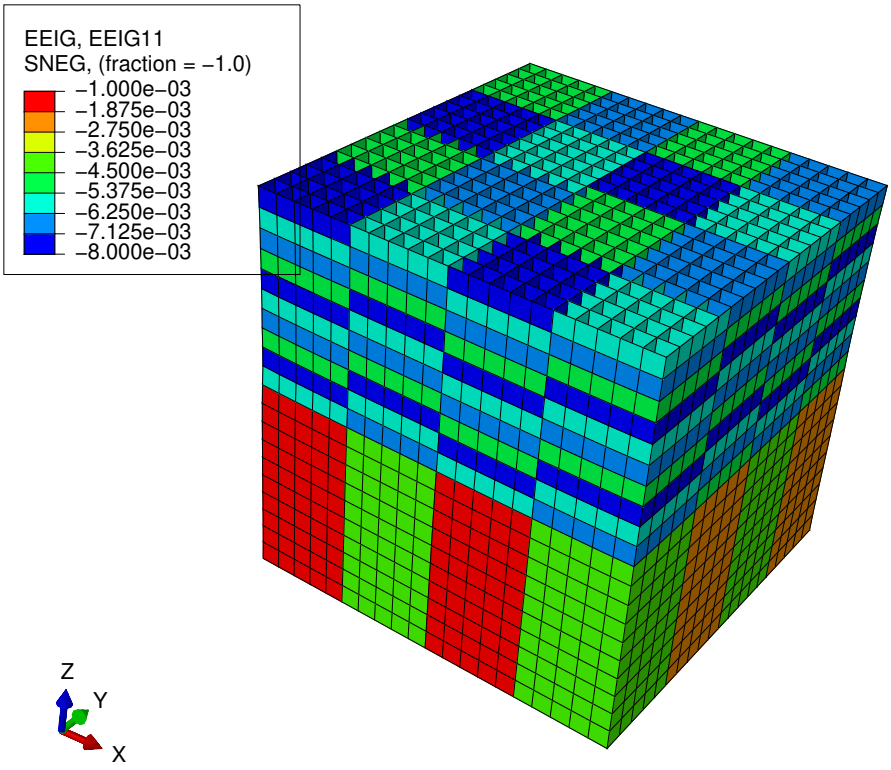


Figure 4: Quilt-style contour plots of eigenstrain component EEIG11 (S4 elements).

Trajectory-based eigenstrain analysis

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Eigenstrain analysis using a fixed tool orientation

Elements tested

C3D8

Features tested

Trajectory-based eigenstrain approach for additive manufacturing simulation with the tool orientation fixed in the Z-direction and fixed deposition bead height and width.

Problem description

This model demonstrates how Abaqus/Standard uses the trajectory-based eigenstrain method for additive manufacturing simulations. It assumes a fixed tool orientation along the Z-direction during the simulation. The height and width of the deposition bead are also fixed.

The geometry of this model consists of two distinct parts: a rectangular frame that includes all the elements to be activated and a fixed rigid part acting as a support to the elements to be activated. The bottom surface of the frame is tied to the top surface of the support part. A linear elastic material definition with an orientation initially aligned to the global coordinate system is assigned to the elements to be activated representing the frame part. The simulation updates the material orientation for the activated elements to align with the tool trajectory so that

the ε^{*}_{11} eigenstrain tensor component is along the tool path.

The model definition follows the steps provided in *"Specifying Element Activation and Eigenstrain Using the Trajectory-Based Method"*.

In the initial configuration, the elements representing the frame are not active. During the simulation, the tool moves along a trajectory following the frame. Two counterclockwise loops stacked in the Z-direction are defined in event series type *"ABQ_AM_HeatSourceTrajectory_RuleID"*. In this event series, the tool is turned off when it is repositioned vertically (that is, in the stack direction) for the second loop. Four eigenstrain tensors are defined in the table collection and the provided event series, allowing a different eigenstrain to be applied to the activated elements on each side of the frame. The deposition bead is positioned below the trajectory defined in the event series.

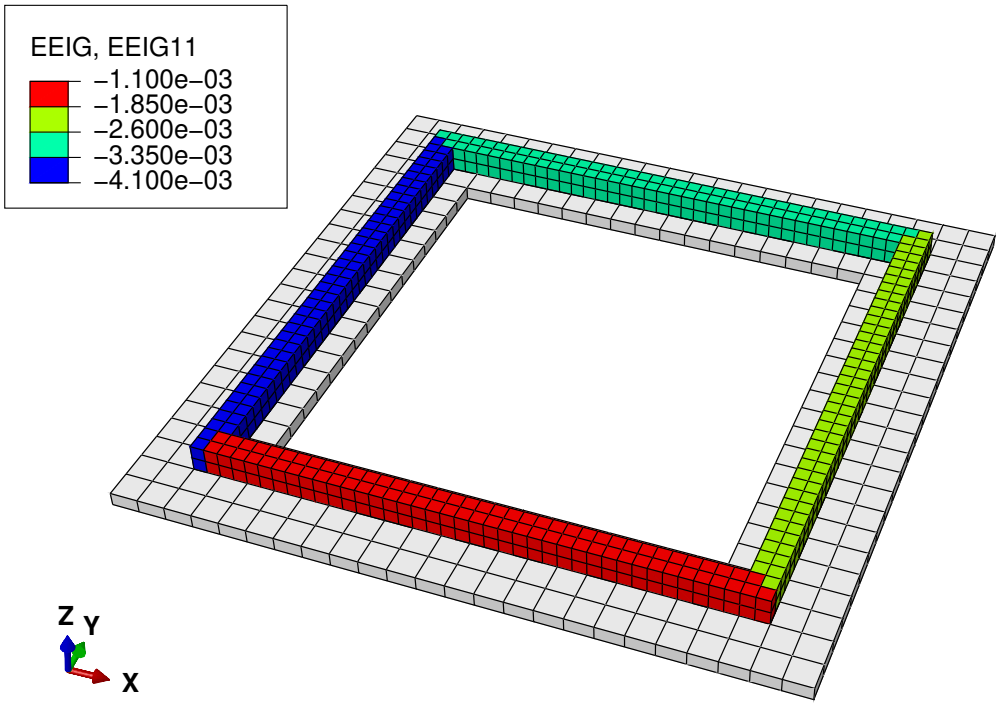


Figure 1: Applied eigenstrain 11-component (along the tool trajectory) in the activated elements.

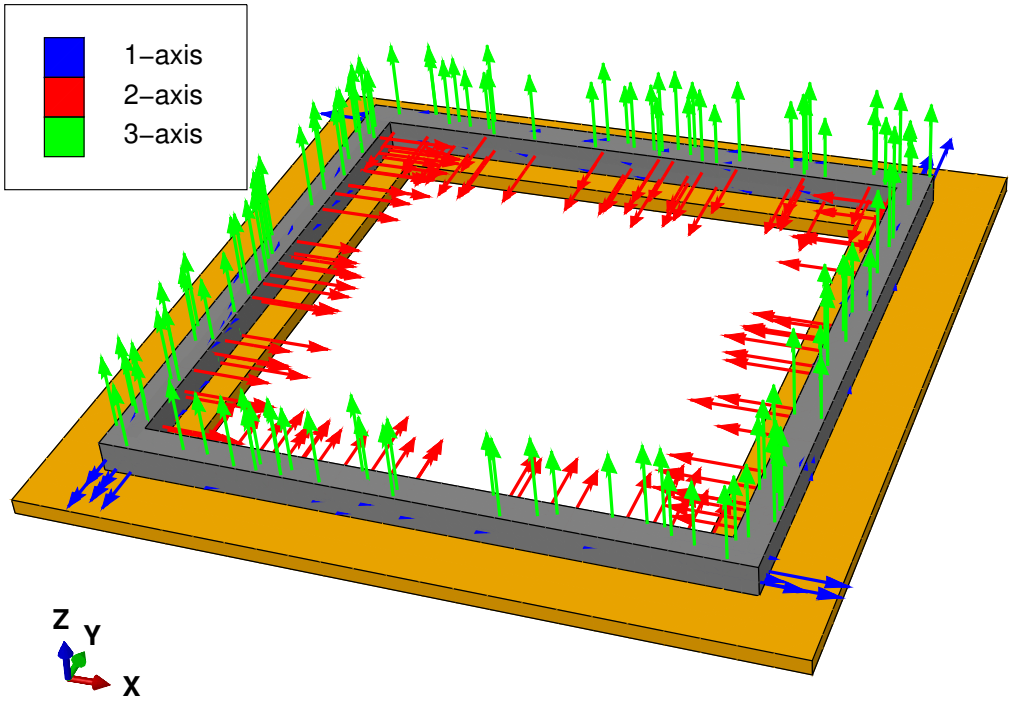


Figure 2: Material orientation after element activation.

Results and discussion

The results provided by running this model indicate that the elements are activated as expected, based on the provided tool trajectory. The applied eigenstrains (once the elements are activated) are consistent with the definitions provided in the table collection used by this model and specification provided in the event series.

Input files

frame_3dprint_with_fixed_z_norm_traj_based_eigen.inp	Model using fixed tool direction and fixed bead dimensions.
ABQ_am_special_purpose_types.inp	Types of property tables, parameter tables, and event series used by the special-purpose techniques for the simulation of common additive manufacturing processes.

Eigenstrain analysis using a time-dependent tool orientation and bead dimensions

Elements tested

C3D20R

Features tested

Trajectory-based eigenstrain approach for additive manufacturing simulation with time-dependent tool orientation and time-dependent deposition bead height and width.

Problem description

This model demonstrates how Abaqus/Standard uses the trajectory-based eigenstrain approach for additive manufacturing simulations. It assumes a time-dependent tool orientation is employed during the simulation. The height and width of the deposition bead are also time-dependent.

The geometry of this model consists of two distinct parts: a three-dimensional frame that includes all the elements to be activated and four fixed rigid supports The bottom corner regions of the frame are tied to the top surfaces of the rigid supports. A linear elastic material definition with an orientation initially aligned to the global coordinate system is assigned to the elements to be activated representing the frame part. The material orientation for the activated elements is updated during simulation to be aligned with the tool trajectory so that the ε^*_{11} eigenstrain tensor component is along the tool path.

The model definition follows the steps provided in *"Specifying Element Activation and Eigenstrain Using the Trajectory-Based Method"*.

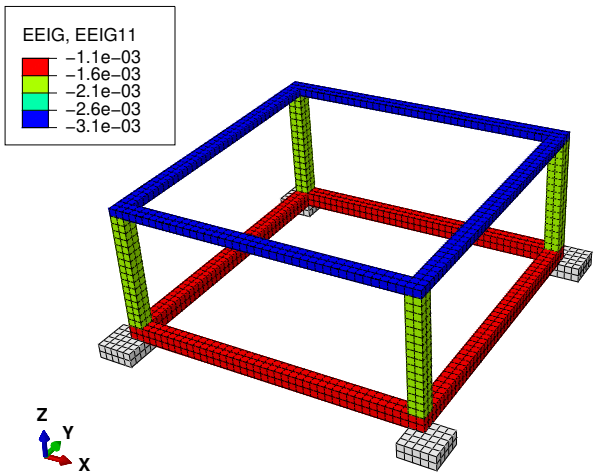


Figure 3: Applied eigenstrain 11-component (along the tool trajectory) in the activated elements.

In the initial configuration, the elements representing the frame are not active. During the simulation, the tool moves along a trajectory defined using event series type *"ABQ_AM_EigenStrain_TrajectoryBased_5AxisStrategy_VariableCrossSection"*. In the event series used in this model, the tool orientation is modified during the activation of the vertical columns and the top portion of the frame. Three different eigenstrain definitions (rules) are used in the model for the bottom layer, vertical columns, and top layer of the frame. The event series also includes time-dependent height and bead dimensions. The deposition bead is centered on the trajectory defined in the event series.

Results and discussion

Results of this model verify that the element activation aligns with the tool trajectory, tool orientation, and bead dimensions specified as part of the event series. The applied eigenstrains, once the elements are active, are consistent with the definitions provided in the table collection used by this model and the specification in the event series.

Input files

frame_3dprint_with_timedep_norm_traj_based_eigen.inp	Model using time-dependent tool orientation and bead dimensions.
ABQ_am_special_purpose_types.inp	Types of property tables, parameter tables, and event series used by the special-purpose techniques for the simulation of common additive manufacturing processes.

Metallurgical phase transformations

Products: Abaqus/Standard

References:

- *Modeling Metallurgical Phase Transformations*

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

DC3D8

Features tested

This section provides basic verification tests of metallurgical phase transformations.

Problem description

This verification test contains single-element heat transfer analyses with nodal temperature degree of freedom driven by predefined amplitude curves. The test is designed to qualitatively verify metallurgical phase transformations under predefined thermal histories.

Material

Two materials, titanium alloy Ti-6AL-4V and steel 5140, are tested.

Titanium alloy Ti-6AL-4V consists primary of three solid phases: a hexagonal close-packed (hcp) -phase, a body-centred cubic (bcc) -phase, and a martensitic ’-phase. The material is considered as fully -phase on solidification. Three transformations can occur between those three solid phases. They can be categorized as one martensitic transformation and two diffusional transformations, as shown in *Table 1*. The solidus and liqudus temperatures of Ti-6AL-4V are 1604°C and 1650°C, respectively.

A simplified metallurgical model of steel 5140 is used. The material consists of five solid phases: austenite(A), bainite (B), ferrite (F), pearlite (P), and martensite (M). The material is considered as fully austenite on solidification. Eight transformations can occur between those five solid phases. They can be categorized as one martensitic and seven diffusional transformations, as shown in *Table 2*. The solidus and liqudus temperatures of steel 5140 are 1750°C and 1800°C, respectively.

Boundary conditions

The nodal temperature degree of freedom is driven by predefined amplitude curves.

Two different thermal histories are applied to test metallurgical phase transformations in titanium alloy Ti-6Al-4V. One thermal history contains thermal cycles ranging between 26°C and 1826°C with constant heating or cooling rates. The other thermal history is a typical thermal cycle experienced during an additive manufacturing process followed by a slow heat treatment process.

One thermal history is applied to test metallurgical phase transformations in steel 5140. The thermal history contains thermal cycles ranging between 20°C and 1826°C.

Results and discussion

The results of metallurgical phase transformations in titanium alloy Ti-6Al-4V with the two thermal cycles are shown in *Figure 1* and *Figure 2*, respectively. Graphs show the temperature, RLS state, and phase content. The RLS indicator describes the material state as Raw (-1), Liquid (0), or Solid (1). It can be seen that the microstructure is fully -phase on solidification. The -phase quickly transforms to '-phase during the rapid cooling that is typical during printing processes. The two diffusional transformations occur slowly during heating and slow cooling conditions. The results agree with the specified material definition.

The results of the metallurgical phase transformations in steel 5140 for a thermal cycle with differenced segments of constant rate are shown in *Figure 3*. At the slow cooling rate (1°C/s), the alloy transforms to mainly ferrite and pearlite. At the medium cooling rate (10°C/s), a small amount of bainite phase is formed. At the high cooling rate (100°C/s), the martensitic phase transformation occurs and only negligible diffusional transformations occur. The results agree with the specified material definition.

Input files

oneelem_phase_trans_ti64.inp	Single-element analysis of metallurgical phase transformations in titanium alloy Ti-6Al-4V by applying temperature cycles with constant heating and cooling rates.
oneelem_phase_trans_ti64_amprint.inp	Single-element analysis of metallurgical phase transformations in titanium alloy Ti-6Al-4V by applying typical thermal cycles of manufacturing processes.
oneelem_phase_trans_s5140.inp	Single-element analysis of metallurgical phase transformations in steel 5140 by applying temperature cycles with constant heating and cooling rate.

Input files containing definitions or data to be included in the input files listed above

ABQ_phase_trans_types.inp	Types of property tables and parameter tables used by the metallurgical phase transformation framework in Abaqus.
amp_temp_amprint.inp	Amplitude curves representing typical thermal cycles during printing and heat treatment processes.
mat_phase_trans_ti64.inp	Material definitions of titanium alloy Ti-6Al-4V.
mat_phase_trans_s5140.inp	Material definitions of steel 5140.

References

- Xie, J., V. Oancea, and J. Hurtado, "Phase Transformations in Metals During Additive Manufacturing Processes," *NAFEMS World Congress*, 2017.
- Zhang, Q., J. Xie, Z. Gao, T. London, D. Griffiths, and V. Oancea, "A Metallurgical Phase Transformation Framework Applied to SLM Additive Manufacturing Processes," *Materials & Design*, vol. 166 107618, 2019.

Tables

Table 1: Phase transformations considered in the metallurgical model of Ti-6Al-4V.

Transformation		Type	Occurring conditions	
1	$\beta \rightarrow \alpha'$	Martensitic	$\dot{\theta}$ < –410°C/s	θ < 650°C
2	$\alpha' \rightarrow \alpha + \beta$	Diffusional	$\dot{\theta}$ > 0°C/s	None
3	$\beta \leftrightarrow \alpha$	Diffusional	$\dot{\theta}$ > –410°C/s	None

Table 2: Phase transformations considered in the metallurgical model of steel 5140.

Transformation		Type	Occurring conditions	
1	$A \rightarrow F$	Diffusional	$\dot{\theta}$ < 0°C/s	539°C < θ < 735°C
2	$A \rightarrow P$	Diffusional	$\dot{\theta}$ < 0°C/s	539°C < θ < 735°C
3	$A \rightarrow B$	Diffusional	$\dot{\theta}$ < 0°C/s	333°C < θ < 539°C
4	$A \rightarrow M$	Martensitic	$\dot{\theta}$ < 0°C/s	θ < 333°C
5	$F \rightarrow A$	Diffusional	$\dot{\theta}$ > 0°C/s	θ > 735°C
6	$P \rightarrow A$	Diffusional	$\dot{\theta}$ > 0°C/s	θ > 735°C
7	$B \rightarrow A$	Diffusional	$\dot{\theta}$ > 0°C/s	θ > 735°C
8	$M \rightarrow A$	Diffusional	$\dot{\theta}$ > 0°C/s	θ > 735°C

Figures

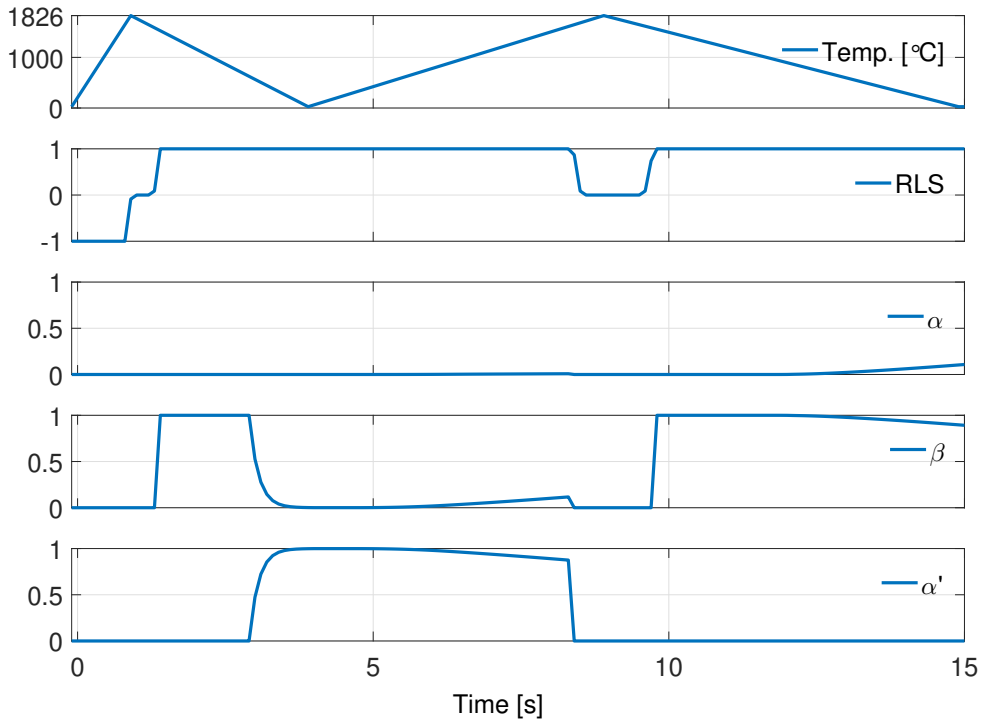
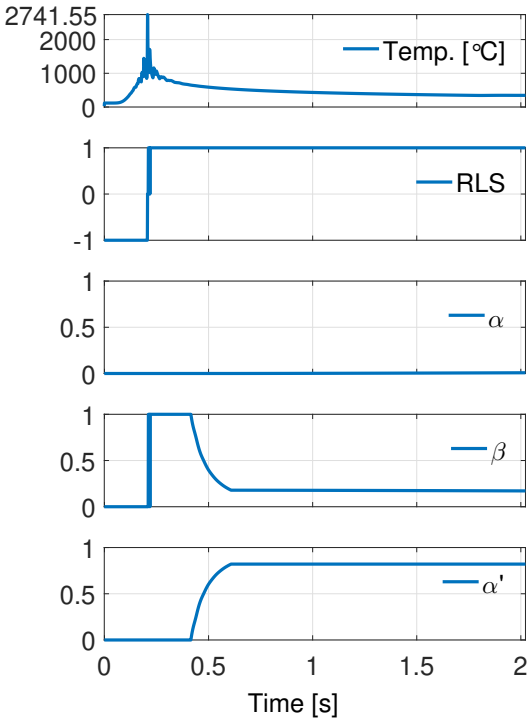


Figure 1: Ti-6Al-4V metallurgical phase transformations under temperature cycles with constant heating and cooling rates.

(a) During printing



(b) During heat treatment

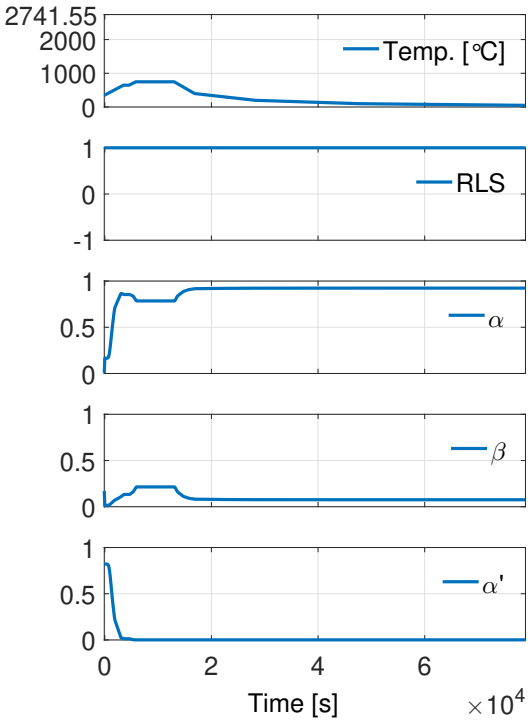
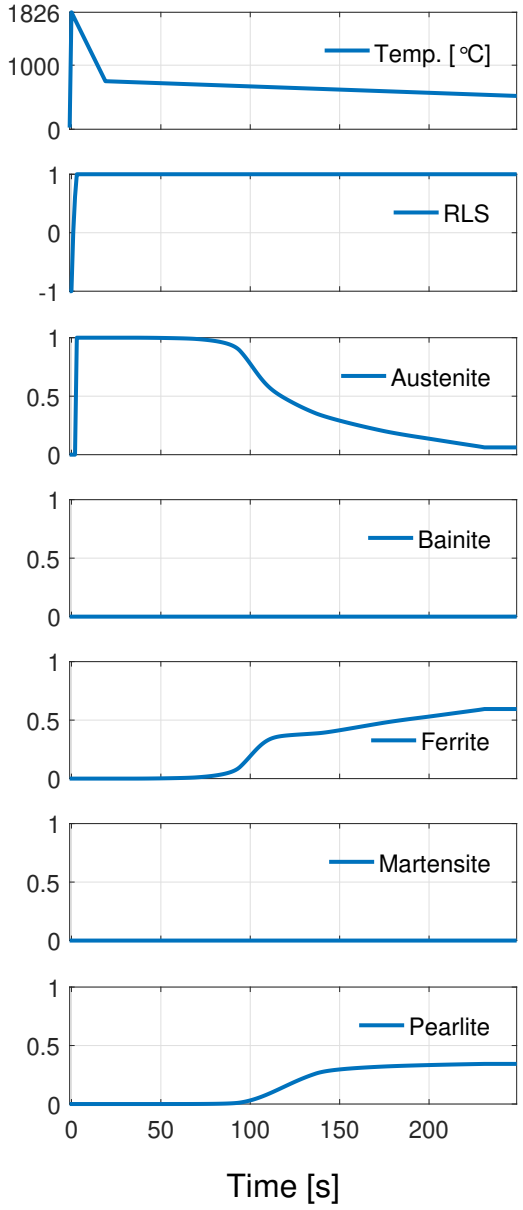
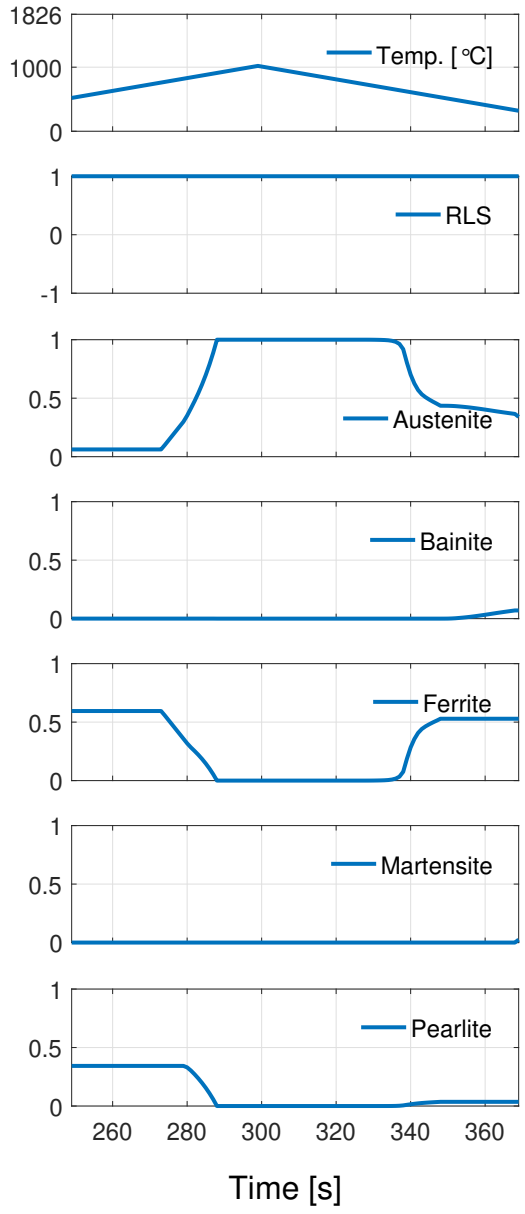


Figure 2: Ti-6Al-4V metallurgical phase transformations under typical temperature cycles of manufacturing processes.

(a) Cooling rate 1°C/s



(b) Cooling rate 10°C/s



(c) Cooling rate 100°C/s

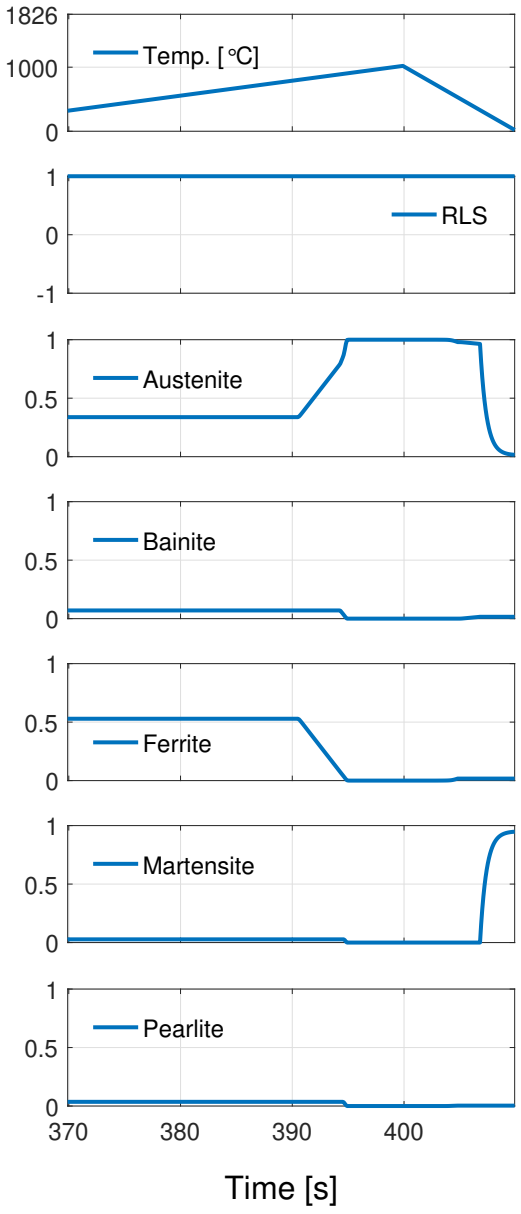


Figure 3: Steel 5140 metallurgical phase transformations under temperature cycles with constant heating and cooling rates.

Material deposition

Products: Abaqus/Standard

References:

- *Special-Purpose Techniques for Additive Manufacturing*

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

DC3D8

Problem description

This section illustrates some of the options available for material deposition in an FDM- and LDED-type additive manufacturing process. The simple model geometry was chosen to clearly depict the motion and orientation of the bead as it deposits material.

Model The model is a $10 \times 10 \times 10$ cube.

Mesh A uniform $20 \times 20 \times 20$ mesh of DC3D8 elements is used.

Material All cubes have the same constant thermal properties.

Conductivity	k = 5.70496e-3
Density	ρ = 4.43e-09
Specific heat	c_p = 543917000

(The units are not important.)

Boundary conditions Each cube is insulated on all faces. An initial temperature of 300° is prescribed for all elements.

Loading No loads are applied.

Results and discussion

In all cases, the material deposition patterns are as expected. The volume fraction output variable (EACTIVE) is used to track the material deposition pattern.

Input files

matDep5AxisAbove.inp	Material deposition simulation using a rotating bead with the bead offset above the tool path. DC3D8 elements.
matDep5AxisMiddle.inp	Material deposition simulation using a rotating bead with the bead set to the middle of the tool path. DC3D8 elements.

matDep5AxisBelow.inp	Material deposition simulation using a rotating bead with the bead offset below the tool path. DC3D8 elements.
matDep5AxisBelow_withTilt.inp	Material deposition simulation using a nonrotating tilted bead with the bead offset below the tool path. DC3D8 elements.
matDep5AxisVarBead.inp	Material deposition simulation using a rotating bead that varies in height and width along the tool path. DC3D8 elements.
matDepBeadOri.inp	Material deposition simulation using a nonrotating bead that is oriented at a fixed angle relative to the global *x*- and *y*-axes along the tool path. DC3D8 elements.
matDepNoRotVarBead.inp	Material deposition simulation using a nonrotating bead that varies in height and width along the tool path. DC3D8 elements.
matDepNoRotXStackDir.inp	Material deposition simulation using a nonrotating bead with the height (stack direction) aligned with the global *x*-direction. DC3D8 elements.
matDepNoRotYStackDir.inp	Material deposition simulation using a nonrotating bead with the height (stack direction) aligned with the global *y*-direction. DC3D8 elements.
elemsAM.inp	Element connectivity for material deposition simulations.
nodesAM.inp	Node definitions for material deposition simulations.
ABQ_am_special_purpose_types.inp	Include file containing type definitions of parameter tables, property tables, and event series definitions.

Moving heat sources

Products: Abaqus/Standard

References:

- *Special-Purpose Techniques for Additive Manufacturing*

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

DC3D8

Problem description

This section illustrates some of the options available for moving uniform heat sources in an additive manufacturing process. The simple model geometry and material properties were chosen to clearly depict the motion and orientation of the heat source as it moves through the model.

Model The model is a $10 \times 10 \times 10$ cube.

Mesh A uniform $20 \times 20 \times 20$ mesh of DC3D8 elements is used.

Material All cubes have the same constant thermal properties. The low thermal conductivity was chosen to slow the diffusion of heat in the model so that it is easier to see the path of the heat source.

Conductivity	k = 5.70496e-3
Density	ρ = 4.43e-09
Specific heat	c_p = 543917000

(The units are not important.)

Boundary conditions Each cube is insulated on all faces. An initial temperature of 300° is prescribed for all elements.

Loading A moving uniform heat source is applied to the model.

Results and discussion

In all cases, the heating patterns are as expected. The temperature output variable (NT11) is used to track the thermal pattern.

Input files

movHeat5AxisVarBead.inp	Transient heat transfer simulation with a rotating heat source and a varying height and width moving through the cube. DC3D8 elements.
movHeat5AxisNoRot_noTilt.inp	Transient heat transfer simulation with a nonrotating, nontilted uniform heat source moving through the cube. DC3D8 elements.

movHeat5AxisNoRot_withTilt.inp	Transient heat transfer simulation with a nonrotating, tilted uniform heat source moving through the cube. DC3D8 elements.
movHeat5Axis.inp	Transient heat transfer simulation with a rotating uniform heat source moving through the cube. DC3D8 elements.
movHeatNoRotVarBead.inp	Transient heat transfer simulation with a nonrotating heat source and a varying height and width moving through the cube. DC3D8 elements.
elemsAM.inp	Element connectivity for material deposition simulations.
nodesAM.inp	Node definitions for material deposition simulations.
ABQ_am_special_purpose_types.inp	Include file containing type definitions of parameter tables, property tables, and event series definitions.

Material removal

Products: Abaqus/Standard

References:

- *Special-Purpose Techniques for Additive Manufacturing*

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

DC3D8

Problem description

This section illustrates some of the options available for material removal for a machining process. The simple model geometry was chosen to clearly depict the motion and orientation of the bead as it removes material.

Model The model is a $10 \times 10 \times 10$ cube.

Mesh A uniform $20 \times 20 \times 20$ mesh of DC3D8 elements is used.

Material All cubes have the same constant thermal properties.

Conductivity	$k = 5.70496\text{e-}3$
Density	$\rho = 4.43\text{e-}09$
Specific heat	$c_p = 543917000$

(The units are not important.)

Boundary conditions Each cube is insulated on all faces. An initial temperature of 300° is prescribed for all elements.

Loading No loads are applied.

Results and discussion

In all cases, the material removal patterns are as expected. The volume fraction output variable (EACTIVE) is used to track the material removal.

Input files

matRemBeadOri.inp	Material removal simulation using a nonrotating bead that is oriented at a fixed angle relative to the global *x*- and *y*-axes along the tool path. DC3D8 elements.
matRem5AxisMiddle.inp	Material removal simulation using a rotating bead with the bead set to the middle of the tool path. DC3D8 elements.

matRemNoRotYStackDir.inp	Material removal simulation using a nonrotating bead with the height (stack direction) aligned with the global *y*-direction. DC3D8 elements.
matRemNoRotVarBead.inp	Material removal simulation using a nonrotating bead that varies in height and width along the tool path. DC3D8 elements.
elemsAM.inp	Element connectivity for material deposition simulations.
nodesAM.inp	Node definitions for material deposition simulations.
ABQ_am_special_purpose_types.inp	Include file containing type definitions of parameter tables, property tables, and event series definitions.

Electromagnetic analysis

In this section:

- *Eddy current analysis*
- *Magnetostatic analysis*

Eddy current analysis

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

EMC2D3 EMC2D4 EMC3D4 EMC3D8

Features tested

Time-harmonic and transient responses of eddy current boundary value problems with excitations due to volume or body current density $\mathbf{J}$ or surface current density $\mathbf{K}$.

Problem description

Two types of problems are solved corresponding to two types of excitations. Both problems result in a constant magnetic flux density $\mathbf{B}$ in the domain. The input files with body current excitation $\mathbf{J}$ are categorized as CCBL (constant curl body load) problems, and the input files with surface current excitation $\mathbf{K}$ are categorized as CCSC (constant curl surface current) problems.

CCBL problems: Only time-harmonic eddy current problems have been tested within this category. The domain in the two-dimensional problems is a square lying in the first quadrant of the plane; in the three-dimensional problems the domain is a cuboid lying in the first octant in space. For the differential equation $\nabla \times \left(\mu^{-1}\nabla \times \mathbf{A}\right) + i\omega\sigma^{E}\mathbf{A} = \mathbf{J}$, the solution sought is $\mathbf{A} = \mathbf{A}_R + i\mathbf{A}_I$, where $\mathbf{A}_R = \mathbf{A}_I = -y\,\mathbf{e}_1 + x\,\mathbf{e}_2$. For this solution the first term in the differential equation vanishes. Therefore, a nonuniform body load (CJNU) of $\mathbf{J} = i\omega\sigma^{E}\mathbf{A}$ is applied everywhere in the domain. Nonzero boundary conditions (distributed surface magnetic vector potential) on the outer boundary and symmetry boundary conditions on the symmetry planes are also specified.

CCSC problems: Both time-harmonic and transient eddy current problems have been tested within this category. The domain for some of the two-dimensional problems is a quarter of a circle lying in the first quadrant of the plane; in the three-dimensional problems the domain is a quarter of a cylinder lying in the first octant in space, with the axis of the cylinder aligned along the global Z-direction. Surface current loads $\mathbf{K} = (2/\mu)\,\mathbf{e}_\theta$ are specified on the outer boundary as a Neumann-type boundary condition. Symmetry boundary conditions are specified on the symmetry planes. The analytical solution in this case is $\mathbf{A}_R = \mathbf{A}_I = r\,\mathbf{e}_\theta$ for the time-harmonic problem, which is the same as that of the CCBL problems. The solution (real only) for the transient problem is identical.

The problems testing the transient eddy current procedure have similar domains (except for the two-dimensional problems with input file names beginning with ccsc_2d_, which consist of stand-alone electromagnetic elements subjected to boundary conditions/loading). In some of the problems the magnetic properties are defined to be different in different regions of the model; in particular, linear properties are used in one region while nonlinear properties are used in another region. The surface current loading results in a constant magnetic field within the

domain, but the magnetic flux density varies based on the material behavior. Both isotropic and orthotropic magnetic behavior have been tested.

A few problems also test motional effects on the solution. In all cases a uniform translational velocity is applied. The magnetic field remains the same as the problem without motion, but the electric fields are modified due to the motional effects.

Material properties: Magnetic permeability of $\mu = 1.25663706144 \times 10^{-6}$ H/m or N/A^2 for free space is used throughout for all the time-harmonic problems and in the regions with linear magnetic behavior for the transient problems. For regions with nonlinear magnetic behavior, the response is defined in terms of a B–H curve describing the strength of the magnetic flux density as a function of the strength of the magnetic field. *Table 1* provides the B–H curve used in these tests.

Table 1: Nonlinear B–H response.

B	H
0	0
1000	7.9577×10^5
1500	1.5915×10^6
1700	2.3873×10^6

A small electrical conductivity (compared to that of a metal) of σ^E = 1.0 or 0.58 S/m is used.

Excitation frequency for time-harmonic problems: $\omega = 2\pi f$ rad/s, $f =$ 50 or 60 Hz.

Loading for transient problems: Transient problems are loaded with a surface current magnitude that varies with time. Some of the problems use a sinusoidal time variation.

Results and discussion

For the time-harmonic problems, the results $\mathbf{B} = 2(1+i)\,\mathbf{e_3}$ and $\mathbf{E} = -i\omega\mathbf{A}$ are verified for all the problems everywhere in the domain. For the transient problems, the solutions (after a number of cycles) are typically verified against corresponding time-harmonic solutions; the solutions are often verified by hand calculations.

Input files

Time-harmonic problems

ccbl_8emc2d3_rnd.inp	8 EMC2D3 elements with nonuniform $\mathbf{J}$.
ccbl_8emc2d3_rnd.f	User subroutine *UDECURRENT* used in ccbl_8emc2d3_rnd.inp.
ccbl_4emc2d4_rnd.inp	4 EMC2D4 elements with nonuniform $\mathbf{J}$ in a cylindrical system and temperature-dependent material properties.
ccbl_4emc2d4_rnd.f	User subroutine *UDECURRENT* used in ccbl_4emc2d4_rnd.inp.
ccbl_24emc3d4_rnd.inp	24 EMC3D4 elements with nonuniform $\mathbf{J}$.
ccbl_24emc3d4_rnd.f	User subroutine *UDECURRENT* used in ccbl_24emc3d4_rnd.inp.
ccbl_4emc3d8_rnd.inp	4 EMC3D8 elements with nonuniform $\mathbf{J}$.

ccbl_4emc3d8_rnd.f	User subroutine *UDECURRENT* used in ccbl_4emc3d8_rnd.inp.
ccbl_8emc3d6_rnd.f	User subroutine *UDECURRENT* used in ccbl_8emc3d6_rnd.inp.
ccbl_200emc2d3_rnd.inp	200 EMC2D3 elements with nonuniform **J**.
ccbl_200emc2d3_rnd.f	User subroutine *UDECURRENT* used in ccbl_200emc2d3_rnd.inp.
ccbl_100emc2d4_reg.inp	100 EMC2D4 elements with nonuniform **J**.
ccbl_100emc2d4_reg.f	User subroutine *UDECURRENT* used in ccbl_100emc2d4_reg.inp.
ccbl_100emc3d8_reg.inp	100 EMC3D8 elements with nonuniform **J**.
ccbl_100emc3d8_reg.f	User subroutine *UDECURRENT* used in ccbl_100emc3d8_reg.inp.
ccbl_4emc2d4_rnd_mot.f	Same setting as in ccbl_4emc2d4_rnd.inp, with the addition of translational velocity.
ccbl_200emc2d3_rnd_mot.f	Same setting as in ccbl_200emc2d3_rnd.inp, with the addition of translational velocity.
ccbl_24emc3d4_rnd_mot.inp	Same setting as in ccbl_24emc3d4_rnd.inp, with the addition of translational velocity.
ccbl_100emc3d8_reg_mot.inp	Same setting as in ccbl_100emc3d8_reg.inp, with the addition of translational velocity.
ccsc_emc2d3.inp	EMC2D3 elements with circumferentially uniform **K** on the outer boundary.
ccsc_emc2d3_nu.inp	EMC2D3 elements with circumferential **K** specified on the outer boundary using user subroutine *UDSECURRENT*.
ccsc_emc2d3_nu.f	User subroutine *UDSECURRENT* used in ccsc_emc2d3_nu.inp.
ccsc_emc2d3_ortho.inp	EMC2D3 elements with circumferentially uniform **K** on the outer boundary and orthotropic material properties.
ccsc_triquad.inp	EMC2D3 and EMC2D4 elements with circumferentially uniform **K** on the outer boundary.

Transient problems

ccsc_solenoid_hex8_tdsine.inp	EMC3D8 elements with circumferentially uniform **K** on the outer boundary.
ccsc_solenoid_tet4_tdsine.inp	EMC3D4 elements with circumferentially uniform **K** on the outer boundary.
ccsc_solenoid_tet4_td_stb.inp	EMC3D4 elements with circumferentially uniform **K** on the outer boundary; also uses stabilization.
ccsc_2d_nlbh_td.inp	EMC2D3 and EMC2D4 elements with uniform **K** on the outer boundary.
ccsc_2d_nlbh_tdsine.inp	EMC2D3 and EMC2D4 elements with uniform **K** on the outer boundary.
ccsc_2d_nlbh_td_temp_dep.inp	EMC2D3 and EMC2D4 elements with uniform **K** on the outer boundary; temperature-dependent material properties.
ccbl_4emc2d4_rnd_tr_mot.inp	Same general setting as ccbl_4emc2d4_rnd.inp, but a transient analysis with translational velocity specified. The analysis assumes the same general form of the magnetic vector potential as ccbl_4emc2d4_rnd.inp, but one that is exponentially decreasing with time. User subroutines *UDECURRENT* and *UDEMPOTENTIAL* are utilized.

Magnetostatic analysis

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

EMC2D3 EMC2D4 EMC3D4 EMC3D8

Features tested

Magnetostatic response of electromagnetic domains subjected to surface current density $\mathbf{K}$.

Problem description

The problems involve rectangular or cylindrical domains that are subjected to an impressed surface current density at the boundary such that the resulting magnetic field is uniform. In one of the models the magnetic properties are nonuniform (different magnetic behavior in different regions of the model); as a result, the magnetic flux density is correspondingly nonuniform. Both linear and nonlinear material properties are used for the nonuniform case.

Material properties: Magnetic permeability of $\mu = 1.25663706144 \times 10^{-6}$ H/m or N/A^2 for free space is used in the regions with linear magnetic behavior. For regions with nonlinear magnetic behavior, the response is defined in terms of a B–H curve describing the strength of the magnetic flux density as a function of the strength of the magnetic field. *Table 1* provides the B–H curve used in these tests.

Table 1: Nonlinear B–H response.

B	H
0	0
1000	7.9577×10^5
1500	1.5915×10^6
1700	2.3873×10^6

Results and discussion

The magnetic field is uniform in the domain. The magnetic flux density is appropriately nonuniform for the problem with nonuniform magnetic material behavior.

Input files

ccsc_4emc2d4_rnd.inp	Rectangular domain with four EMC2D4 elements with uniform $\mathbf{K}$ on the boundary.

ccsc_solenoid_tet4_mqs_stb.inp	Quarter-cylindrical domain with EMC3D4 elements with uniform $\mathbf{K}$ on the boundary.

Coupled thermal-electrochemical analysis

In this section:

- *Fully coupled thermal-electrochemical procedures*
- *Modeling a half-cell lithium metal anode battery with multiphysics Butler-Volmer surface loading*
- *Modeling a solid electrolyte and solid-state battery*

Fully coupled thermal-electrochemical procedures

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

QEC3D8 QEC3D8R

Features tested

This verification problem tests the thermal-electrochemical behavior of a lithium ion battery that involves the solution of lithium ion diffusion in the electrolyte and the electrodes, as well as the charge transfer process as captured by the 3D porous electrode theory Newman formulation. All associated thermal effects such as heat losses, temperature evolutions, and temperature-dependent properties are considered in the test cases.

Problem description

The problems test the discharge and charge analyses of a 7.5 Ah Kokam pouch cell as described in *Ecker et al. (2015) (Part I)* and *Ecker et al. (2015) (Part II)*. Discharge analysis is performed at four different current (C) rates, while the charging analysis is done for 1C only, all at an ambient temperature of 25°C. In all cases, the discharge (or charge) curves predicted by the simulation are compared against the experimental results appearing in *Ecker et al. (2015) (Part II)*.

Model: The geometric and other relevant parameters, including individual electrode thicknesses, of the cell are as described in *Ecker et al. (2015) (Part II)*. The total length, width, and thickness of the electrode stack are 0.101, 0.085, and 0.000147 meters, respectively.

Mesh: The finite element model consists of 25 elements each of type QEC3D8 or QEC3D8R through the thickness for the active electrodes (anode and cathode) and 11 elements of the same type to represent the separator. Only one element is used in the plane because the setup is essentially one dimensional. The QEC3D8 and QEC3D8R elements support thermal-electrochemical degrees of freedom.

Material: The anode material is graphite, and the cathode is made up of Li(Ni0.4Co0.6)O2, the thermal-electrochemical properties of which are described in *Ecker et al. (2015) (Part I)* and used in the models included in this section. An Arrhenius-type temperature dependence is used for several of the properties.

Loading: The battery modeling is subjected to currents that correspond to a 1C charge analysis and various rates of discharge ranging from 0.25C to 5C.

Results and discussion

Figure 1 displays the discharge curves predicted by Abaqus showing the evolution of the voltage difference across the cell during discharge as a function of capacity for four different C rates (0.25C, 1C, 3C, and 5C) and compared with experimental results from *Ecker et al. (2015) (Part II)* for the same C rates. A reasonably good correlation is observed. *Figure 2* shows the results from a charging simulation for 1C rate, which also compares

well with the experimental values from *Ecker et al. (2015) (Part II)*. The choice of swelling tables for the charge analysis with swelling resulted in voltage curves that are identical to the 1C rate charge voltage curve.

Input files

ecker-main-input_1C_charge.inp	Main input file for 1C charge analysis with QEC3D8 elements.
ecker-main-input_1C_charge_QEC3D8R.inp	Main input file for 1C charge analysis with QEC3D8R elements.
ecker-main-input_1C_charge_swell.inp	Main input file for 1C charge analysis in a fully coupled thermal-electrochemical-structural procedure with QEC3D8 elements.
ecker-full-input_1C_charge_swell_pore.inp	Complete input file to run a 1C charge analysis in a coupled thermal-electrochemical-structural—pore pressure procedure with QEC3D8 elements.
ecker-main-input_1C_discharge.inp	Main input file for 1C discharge analysis with QEC3D8 elements.
ecker-main-input_1C_discharge_QEC3D8R.inp	Main input file for 1C discharge analysis with QEC3D8R elements.
ecker-main-input_3C_discharge.inp	Main input file for 3C discharge analysis with QEC3D8 elements.
ecker-main-input_3C_discharge_QEC3D8R.inp	Main input file for 3C discharge analysis with QEC3D8R elements.
ecker-main-input_5C_discharge.inp	Main input file for 5C discharge analysis with QEC3D8 elements.
ecker-main-input_5C_discharge_QEC3D8R.inp	Main input file for 5C discharge analysis with QEC3D8 elements.
ecker-main-input_p25C_discharge.inp	Main input file for 0.25C discharge analysis with QEC3D8 elements.
ecker-main-input_p25C_discharge_QEC3D8R.inp	Main input file for 0.25C discharge analysis with QEC3D8R elements.

Input files containing definitions or data to be included in the input files listed above

ABQ_EchemPET_Types.inp	Property and parameter table types.

Input files containing definitions or data to be included in the input files listed above for the charge analysis

ecker-anode_propParam-charge.inp	Anode properties for charge analysis.
ecker-cathode_propParam-charge.inp	Cathode properties for charge analysis.
ecker-electrolyte_propParam-charge.inp	Electrolyte properties for charge analysis.
ecker-separator_propParam-charge.inp	Separator properties for charge analysis.

Input files containing definitions or data to be included in the input files listed above for the charge analysis with swelling.

ecker-anode_propParam-charge_swell.inp	Anode properties for charge analysis with swelling.
ecker-cathode_propParam-charge_swell.inp	Cathode properties for charge analysis with swelling.

ecker-electrolyte_propParam-charge.inp Electrolyte properties for charge analysis.

ecker-separator_propParam-charge_swell.inp Separator properties for charge analysis with swelling.

User subroutine to be included for the charge analysis

ecker-uamp-charge.f User subroutine to stop the analysis when the battery is fully charged.

Input files containing definitions or data to be included in the input files listed above for the discharge analysis

ecker-anode_propParam-discharge.inp Anode properties for discharge analysis.

ecker-cathode_propParam-discharge.inp Cathode properties for discharge analysis.

ecker-electrolyte_propParam-discharge.inp Electrolyte properties for discharge analysis.

ecker-separator_propParam-discharge.inp Separator properties for discharge analysis.

User subroutine to be included for the discharge analysis

ecker-uamp-discharge.f User subroutine to stop the analysis when the battery is fully discharged.

References

- Ecker, M., T. K. D. Tran, P. Dechent, S. Kabitz, A. Warnecke, and D. U. Sauer, "Parameterization of a Physico-Chemical Model of a Lithium-Ion Battery - I. Determination of Parameters," *Journal of The Electrochemical Society*, vol. 162 (9), pp. A1836–A1848, 2015.
- Ecker, M., S. Kabitz, I. Laresgoiti, and D. U. Sauer, "Parameterization of a Physico-Chemical Model of a Lithium-Ion Battery - II. Model Validation," *Journal of The Electrochemical Society*, vol. 162 (9), pp. A1849–A1857, 2015.

Figures

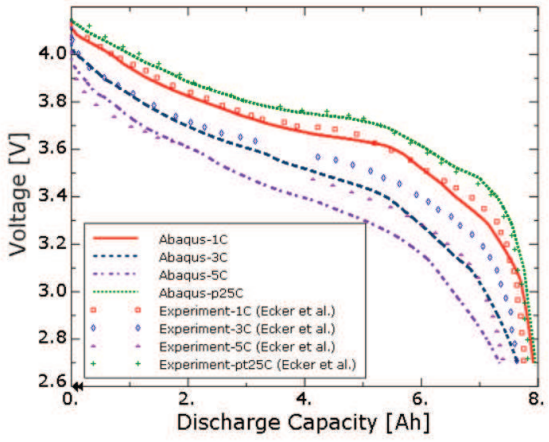


Figure 1: Voltage discharge curves for various C rates.

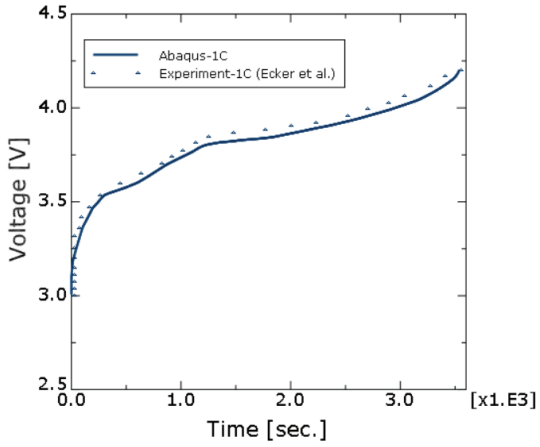


Figure 2: Voltage charge curves for 1C rate.

Modeling a half-cell lithium metal anode battery with multiphysics Butler-Volmer surface loading

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

QEC3D8 QEC3D8R

Features tested

This verification problem tests the thermal-electrochemical behavior of a half-cell battery modeled using a lithium metal anode and a porous cathode.

The lithium ion battery involves the solution of lithium ion diffusion in the electrolyte and the porous electrodes, as well as the charge transfer process that is captured by the 3D formulation of Newman's porous electrode theory. The interaction of the solid anode with the separator is modeled using a Butler-Volmer surface interaction load. All associated thermal effects such as heat losses, temperature evolutions, and temperature-dependent properties are considered in the test cases.

Problem description

The problems test the lithium metal battery cell charging capabilities at different charge rates, as described in *Donnelly Masters thesis* . The results compare very well with that shown in *Donnelly Masters thesis*.

Model: The geometric and other relevant parameters, including individual electrode thicknesses, of the cell are as described in *Donnelly Masters thesis*. The total length, width, and thickness of the electrode stack are 0.101 m, 0.085 m, and 0.000147 m, respectively.

Mesh: The finite element model consists of 44 elements each of type QEC3D8 or QEC3D8R. The solid anode is modeled with one element through the thickness. The porous cathode and separator elements are modeled with 25 and 18 elements, respectively. Only one element is used in the plane because the setup is essentially 1D. The QEC3D8 and QEC3D8R elements support thermal-electrochemical degrees of freedom.

Material: The anode material is solid lithium, and the cathode is made up of lithium cobalt oxide (LCO). The thermal-electrochemical properties of the different regions described in *Donnelly Masters thesis* are used in the models included in this section. An Arrhenius-type temperature dependence is used for several of the properties.

Loading: The battery is subjected to currents that correspond to a 0.01C, 1C, and 2C charge analysis.

Results and discussion

Figure 1 displays the charge curves predicted by Abaqus, showing the evolution of the voltage difference across the cell during charge as a function of time for 0.1C rate and the simulation results from *Donnelly Masters thesis* for the same C rate. A reasonably good correlation is observed.

Input files

Input files to model a half-cell lithium battery with **MULTIPHYSICS LOAD*

solidelectrode_halfcell_pt1C.inp	Main input file for 0.1C charge analysis with QEC3D8 elements.
solidelectrode_halfcell_pt1C_QEC3D8R.inp	Main input file for 0.1C charge analysis with QEC3D8R elements.
solidelectrode_halfcell_1C.inp	Main input file for 1C charge analysis with QEC3D8 elements.
solidelectrode_halfcell_1C_QEC3D8R.inp	Main input file for 1C charge analysis with QEC3D8R elements.
solidelectrode_halfcell_2C.inp	Main input file for 2C charge analysis with QEC3D8 elements.
solidelectrode_halfcell_2C_QEC3D8R.inp	Main input file for 2C charge analysis with QEC3D8R elements.

Input files to model a half-cell lithium battery with **INTERFACE REACTION*

solidelectrode_halfcell_pt1C_Interaction.inp	Main input file for 0.1C charge analysis with QEC3D8 elements.
solidelectrode_halfcell_pt1C_Interaction_QEC3D8R.inp	Main input file for 0.1C charge analysis with QEC3D8R elements.
solidelectrode_halfcell_1C_Interaction.inp	Main input file for 1C charge analysis with QEC3D8 elements.
solidelectrode_halfcell_1C_Interaction_QEC3D8R.inp	Main input file for 1C charge analysis with QEC3D8R elements.
solidelectrode_halfcell_2C_Interaction.inp	Main input file for 2C charge analysis with QEC3D8 elements.
solidelectrode_halfcell_2C_Interaction_QEC3D8R.inp	Main input file for 2C charge analysis with QEC3D8R elements.

References

- Donnelly, C., “Finite Analysis Simulation of a Lithium Metal Battery with Metal Organic Framework (MOF) Layer Using COMSOL Multiphysics,” *Master's Thesis, UCLA*, 2021.

Figures

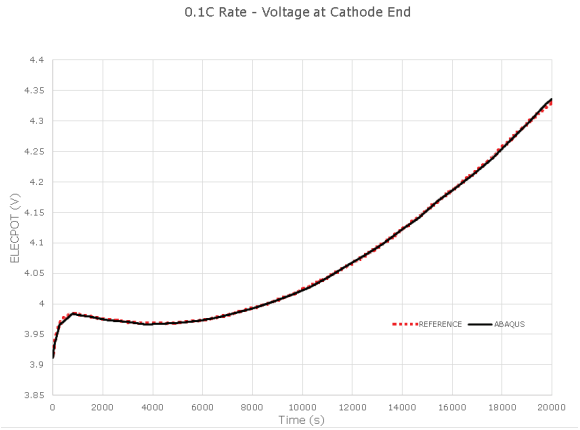


Figure 1: Voltage charge curves for 0.1C rate.

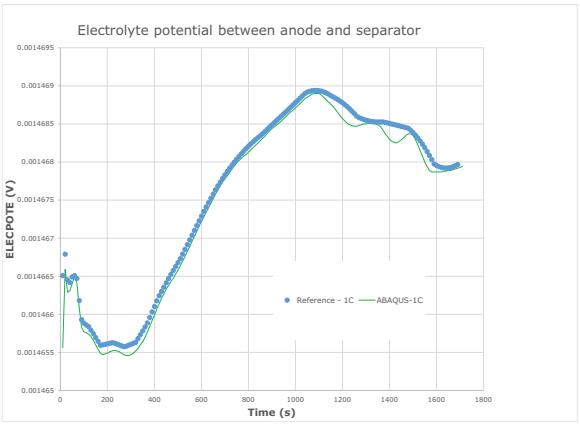


Figure 2: Fluid electric potential curves for 1C rate.

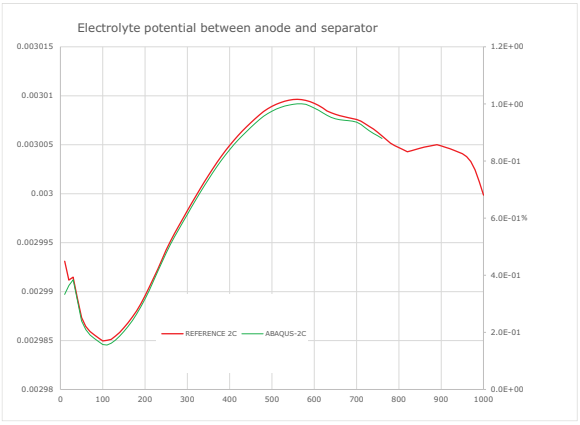


Figure 3: Fluid electric potential curves for 2C rate.

Modeling a solid electrolyte and solid-state battery

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

QEC3D8 QEC3D6 QEC3D4

Features tested

This verification problem tests the thermal-electrochemical behavior of a solid electrolyte battery modeled using a lithium metal anode and a solid cathode.

The lithium ion battery involves the solution of lithium ion diffusion in the separator, as well as the charge transfer process at the interfaces of the anode-separator and cathode-separator. The interaction of the solid electrodes with the separator is modeled using a Butler-Volmer surface interaction load. Concentration of lithium in the cathode is modeled using diffusion process.

Problem description

The problem tests the lithium metal battery cell charging capabilities at different charge rates, as described in *Tian et al*. The results compare very well with that shown in *Tian et al* .

Model: The geometric and other relevant parameters, including individual electrode thicknesses, of the cell are as described in *Tian et al*. The different electrodes are in nanometer lengths. The separator is 1500 nm long, and the cathode is 1820 nm long.

Mesh: The finite element model consists of 3 elements each of type QEC3D8 or QEC3D6 or QEC3D4. The solid anode is modeled with one element through the thickness. The cathode and separator elements are modeled with one element each, respectively. Only one element is used in the plane because the setup is essentially 1D. The QEC3D8, QEC3D6, and QEC3D4 elements support thermal-electrochemical degrees of freedom.

Material: The anode material is solid lithium, and the cathode is made up of lithium cobalt oxide. The thermal-electrochemical properties of the different regions described in *Tian et al* are used in the models included in this section.

Loading: The battery is subjected to a current load that corresponds to a 1C discharge analysis.

Results and discussion

Figure 1 displays the discharge curves predicted by Abaqus, showing the evolution of the voltage difference across the cell during charge as a function of time for 1C rate and the simulation results from for the same C rate. A reasonably good correlation is observed.

Input files

Input files to model a half-cell lithium battery with **MULTIPHYSICS LOAD*

QEC3D4_multiphysicsload.inp	1C discharge analysis with QEC3D4 elements.
QEC3D6_multiphysicsload.inp	1C discharge analysis with QEC3D6 elements.
QEC3D8_multiphysicsload.inp	1C discharge analysis with QEC3D8 elements.

Input files to model a half-cell lithium battery with **INTERFACE REACTION*

qec3d4_surfInterContact.inp	1C discharge analysis with QEC3D4 elements.
qec3d6_surfInterContact.inp	1C discharge analysis with QEC3D6 elements.
qec3d8_surfInterContact.inp	1C discharge analysis with QEC3D8 elements.

References

- Tian, H-K., , and Y. Qi, “Simulation of the Effect of Contact Area Loss in All-Solid-State Li-Ion Batteries,” *Journal of the Electrochemical Society*, vol. 164, no. 11, 2017.

Figures

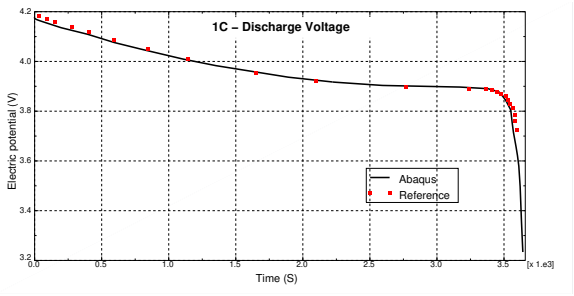


Figure 1: Voltage discharge curves for 1C rate.

Piezoelectric analysis

In this section:

- *Static analysis for piezoelectric materials*
- *Frequency extraction analysis for piezoelectric materials*
- *General analysis procedures for piezoelectric materials*

Static analysis for piezoelectric materials

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

CAX4E

Features tested

The static analysis capability for materials that include piezoelectric coupling is discussed and illustrated. Both mechanical loads and electrical surface charges are applied. In *Mercer, Reddy, and Eve (1987)* a problem subjected to a sinusoidal load is analyzed. The model definition from that problem is used to illustrate the static response due to a constantly applied load. In the following sections the applicable linear dynamics capabilities are discussed.

Problem description

A cylinder of piezoelectric ceramic is subjected to both a pressure load and a distributed charge load. The cylinder is 20 mm thick with an inner radius of 5 mm and an outer radius of 25 mm. The cylinder is subjected in the first step to a pressure load on the top surface. The second step applies a distributed electrical charge on the top surface. Both the top and bottom surfaces have electrodes. The potentials on the bottom surface are prescribed to zero. The electrodes are generated by using equations that set all the potentials to the same value.

The cylinder is modeled as an axisymmetric problem using only one CAX4E element. The material properties for the PZT4 material are given as

Elasticity matrix:

$$\begin{bmatrix} 139.0 & 74.28 & 77.84 & 0 & 0 & 0 \\ 74.28 & 115.4 & 74.28 & 0 & 0 & 0 \\ 77.84 & 74.28 & 139.0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 25.64 & 0 & 0 \\ 0 & 0 & 0 & 0 & 25.64 & 0 \\ 0 & 0 & 0 & 0 & 0 & 25.64 \end{bmatrix} \text{GPa}$$

Piezoelectric coupling matrix (stress coefficients):

$$\begin{bmatrix} 0 & -5.207 & 0 \\ 0 & 15.08 & 0 \\ 0 & -5.207 & 0 \\ 12.710 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 12.710 \end{bmatrix} \text{coulomb/m}^2$$

Dielectric matrix:

$$\begin{bmatrix} 6.752 & 0 & 0 \\ 0 & 5.872 & 0 \\ 0 & 0 & 6.752 \end{bmatrix} \quad 10^{-9} \text{ farad/meter}$$

The material is poled in the 2-direction.

Results and discussion

In the first step the σ_{22} value should be equal and opposite to the applied vertical pressure. It is correctly computed as -1.0×10^6. The stresses in the other directions are negligible. The stress is computed as

$$\sigma_{ij} = D_{ijkl}\varepsilon_{kl} - e^{\varphi}_{mkl}E_m.$$

The σ_{22} term is calculated (neglecting the zero terms) as

$$\sigma_{22} = D_{2211}\varepsilon_{11} + D_{2222}\varepsilon_{22} + D_{2233}\varepsilon_{33} - e^{\varphi}_{222}E_2$$

or

$$\sigma_{22} = \left(7.428 \times 10^{10}\right)\varepsilon_{11} + \left(11.54 \times 10^{10}\right)\varepsilon_{22} + \left(7.428 \times 10^{10}\right)\varepsilon_{33} - (15.08)\,E_2.$$

This relationship can be verified from the results. The electrical flux density is negligible in both directions for the pressure loading. This is correct, considering the flux conservation equation. The potential gradient is constant in the vertical direction. The maximum vertical displacement, -1.65×10^{-7}, occurs at the top surface.

In the second step instead of the pressure load, a distributed electrical charge is applied to the top surface of the model. The q_2 value should be equal and opposite to the charge density applied to the top surface. It is correctly computed as -1.0×10^{-3}. The flux density in the other direction is negligible. The flux densities are computed as

$$q_i = e^{\varphi}_{ijk}\varepsilon_{jk} + D^{\varphi}_{ij}E_j.$$

The q_2 term is calculated (neglecting the zero terms) as

$$q_2 = e^{\varphi}_{211}\varepsilon_{11} + e^{\varphi}_{222}\varepsilon_{22} + e^{\varphi}_{233}\varepsilon_{33} + D^{\varphi}_{22}E_2$$

or

$$q_2 = (-5.207)\,\varepsilon_{11} + (15.08)\,\varepsilon_{22} + (-5.207)\,\varepsilon_{33} + \left(5.872 \times 10^{-9}\right)E_2.$$

This relationship can be verified from the results. This problem, from equilibrium considerations, should produce a stress-free state. The strain field is such that the equation given above for the stress gives a negligible value.

Input files

ppzostat.inp	Input file for this analysis.

References

- Mercer, C. D., B. D. Reddy, and R. A. Eve, “Finite Element Method for Piezoelectric Media,” *UCT/CSIR Applied Mechanics Research Unit Technical Report*, no. 92, April 1987.

Frequency extraction analysis for piezoelectric materials

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

CAX4E CAX8E CPE4E

Features tested

The frequency extraction analysis capability for materials that include piezoelectric coupling is illustrated.

Problem description

The model is the cylinder described in *Static analysis for piezoelectric materials*. Three analyses are performed using two different models. One model uses sixteen CAX4E elements, and the other uses four CAX8E elements. In addition, a single test that extracts the eigenvalues of an unconstrained CPE4E element using temperature- and field-variable-dependent piezoelectric and dielectric properties is included.

Results and discussion

The first 10 modes are extracted. The lowest frequency for the CAX4E element model is 41.8 kHz. The lowest frequency for the CAX8E element model is 44.6 kHz. These mode shapes will be used in the following sections for the linear dynamics options. The restart capability will be used for this purpose. For the CAX8E element model, the total force is output on the edge lying on the x-axis.

For the unconstrained eigenvalue extraction test, the results match the corresponding results for an equivalent model where the piezoelectric and dielectric properties are independent of temperature and field variables.

Input files

ppzofrq1.inp	Frequency extraction with CAX4E elements and **PIEZOELECTRIC*, TYPE=S.
ppzofrq1a.inp	Frequency extraction with CAX4E elements and **PIEZOELECTRIC*, TYPE=E.
ppzofrq2.inp	Frequency extraction with CAX8E elements.
ppzofrqr.inp	Restart of frequency extraction with CAX4E elements.
ece4efe1_tfv.inp	Frequency extraction for an unconstrained CPE4E element using temperature- and field-variable-dependent piezoelectric and dielectric properties.

General analysis procedures for piezoelectric materials

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

In this section the general analysis procedures for elements that include piezoelectric coupling are discussed.

Transient dynamic analysis for piezoelectric materials

Elements tested

C3D8E

Features tested

The transient dynamic analysis capability for elements that include piezoelectric coupling is illustrated. Both concentrated nodal electrical charges and potentials are applied in separate analyses.

Problem description

A piezoelectric bar [1 × 1 × 10] is subjected to an electrical potential. The potentials on the longitudinal top surface are prescribed to 1, and the potentials on the longitudinal bottom surface are prescribed to 0. The electrodes are simulated by using equations that set all the potentials on a face to the same prescribed value. The material is polarized in the local 3-direction.

The block is modeled using five C3D8E elements along the length. The material properties for the PZT-5H material that is used in the tests are as follows:

Elasticity properties:

Engineering constants	
E_1	60.61 GPa
E_2	60.61 GPa
E_3	48.31 GPa
ν_{12}	0.289
ν_{13}	0.512
ν_{23}	0.512
G_{12}	23.5 GPa
G_{13}	23.0 GPa
G_{23}	23.0 GPa

Piezoelectric coupling matrix (strain coefficients):

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 741 & 0 \\ 0 & 0 & 0 & 0 & 0 & 741 \\ -274 & -274 & 593 & 0 & 0 & 0 \end{bmatrix} 10^{-12} \quad \text{m/volt}$$

Dielectric matrix for fully constrained material:

$$\begin{bmatrix} 1.505 & 0 & 0 \\ 0 & 1.505 & 0 \\ 0 & 0 & 1.301 \end{bmatrix} 10^{-8} \quad \text{farad/meter}$$

The piezoelectric coefficient matrix $\mathbf{g}^{\varphi}$ and the dielectric matrix $\mathbf{D}^{\varphi(\sigma)}$ for an unconstrained material, which are commonly used electrical properties in the piezoelectric literature, can be expressed in terms of the piezoelectrical properties mentioned above. These relationships are given in *Piezoelectric analysis*. These properties are commonly provided by the manufacturer. For the PZT-5H material the properties are as follows:

Piezoelectric coefficient matrix $\mathbf{g}^{\varphi}$:

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 26.8 & 0 \\ 0 & 0 & 0 & 0 & 0 & 26.8 \\ -9.11 & -9.11 & 19.7 & 0 & 0 & 0 \end{bmatrix} 10^{-3} \quad \text{volt m/N}$$

Dielectric matrix for an unconstrained material $\mathbf{D}^{\varphi(\sigma)}$:

$$\begin{bmatrix} 2.77 & 0 & 0 \\ 0 & 2.77 & 0 \\ 0 & 0 & 3.01 \end{bmatrix} 10^{-8} \quad \text{farad/meter}$$

The tests involve a transient dynamic step in which the potential on the top surface is ramped up to 1 volt in 0.014 seconds and then held constant for the remainder of the step. The results at the end of the step correspond to the static solution.

Results and discussion

The applied electrical potential of 1 volt results in a potential gradient of 1 volt/m. The piezoelectric constants $d^{\varphi}_{3\,33}$ and $g^{\varphi}_{3\,33}$ can be used to estimate the electrical charge per unit area. In the case of an unconstrained material

$$d^{\varphi}_{3\,33} = \frac{\varepsilon_{33}}{\mathbf{E}_3}$$

and

$$g^{\varphi}_{3\,33} = \frac{\varepsilon_{33}}{q_3},$$

where $\mathbf{E}_3$ is the potential gradient and q_3 is the charge density in the local 3-direction. So the charge density q_3 is equal to $d^{\varphi}_{3\,33}/g^{\varphi}_{3\,33} = 3.01 \times 10^{-8}$. The area to which the voltage is applied is 10; therefore, the static reaction charge should be about 3.01×10^{-7}. The results of *ppzodyn1.inp* confirm this reaction charge. In the input file *ppzodyn2.inp* a concentrated nodal electrical charge of 3.01×10^{-7} is applied instead of a potential value of 1 at the top surface. This results in a potential of 1 volt on the top surface.

Input files

ppzodyn1.inp	Dynamic analysis with prescribed potentials.
ppzodyn2.inp	Dynamic analysis with concentrated nodal electrical charges.

Geometrically nonlinear static analysis for piezoelectric materials

Elements tested

C3D20E

Features tested

The geometrically nonlinear static analysis capability for a piezoelectric material is illustrated. A beam clamped at both ends is subjected to a potential that results in a loading that reaches the critical buckling load.

Problem description

A beam of piezoelectric material is clamped at both sides and is subjected to an electrical potential. The beam is 0.4 m long with a width of 0.006 m and a thickness of 0.005 m. The potentials at one end of the beam are prescribed to 500 kvolts, and the potentials at the other end are prescribed to 0 kvolt. The electrodes are simulated by using equations that set the potential of all the nodes at each end of the beam to the same prescribed value. In the first step a small load is applied to the center to induce a small geometric imperfection.

The block is modeled using 20 C3D20E elements. The material properties for PZT-5H, which is used for the simulation, are given in the previous section.

Results and discussion

The critical buckling load for the beam is

$$P = \frac{4\pi^2 EI}{L^2} = 745\text{N},$$

where E is the Young's modulus in the longitudinal direction and I is the appropriate moment of inertia for the beam section. The analysis shows a critical compressive force of 773 N. The compressive force converges to the analytical buckling load with mesh refinement.

Input files

ppzobuckle.inp	Geometrically nonlinear static analysis.

Large rotations in piezoelectric analysis

Elements tested

C3D4E C3D6E C3D8E C3D10E C3D20RE

Features tested

Large rotations for different piezoelectric element types.

Problem description

Five blocks modeled with different piezoelectric element types are subjected to an electrical potential. The potentials at one side are prescribed to 1 volt, and the potentials at the opposite side are prescribed to 0 volt. The

blocks are tied to three orthogonal surfaces to prevent unconstrained rigid body motions but are free to move tangentially with respect to the surfaces. The surfaces are also used to prescribe the rigid body rotations.

Results and discussion

The applied potential gradient remains constant in magnitude but rotates appropriately with the element.

Input files

ppzolarrot.inp	Large rotations with piezoelectric elements.

Validation of piezoelectric material behavior

Elements tested

C3D8E

Features tested

Validation of piezoelectric material properties using a general static analysis.

Problem description

A block of PZT-5H is subjected to different loadings from which the piezoelectric material properties can be validated.

Results and discussion

In the first step the potentials at the two opposite sides in the local 3-direction of the material are prescribed. Sufficient boundary conditions are applied to prevent rigid body motions, but the model is otherwise unconstrained.

The piezoelectric constants $d^{\varphi}_{3\,33} = 593 \times 10^{-12}$ and $g^{\varphi}_{3\,33} = 19.7 \times 10^{-3}$ can be expressed in terms of the strain ε_{33}, the potential gradient $\mathbf{E}_3$, and the charge density q_3 as

$$d^{\varphi}_{3\,33} = \frac{\varepsilon_{33}}{\mathbf{E}_3}$$

and

$$g^{\varphi}_{3\,33} = \frac{\varepsilon_{33}}{q_3}.$$

The piezoelectric constants $d^{\varphi}_{3\,11}$, $d^{\varphi}_{3\,22}$, $d^{\varphi}_{3\,33}$, $g^{\varphi}_{3\,11}$, $g^{\varphi}_{3\,22}$, and $g^{\varphi}_{3\,33}$ are verified by using the numerically obtained values of the strains ε_{11}, ε_{22}, and ε_{33}. The dielectric constant $D^{\varphi(\sigma)}_{33}$ in the local 3-direction for an unconstrained material is given by

$$D^{\varphi(\sigma)}_{33} = \frac{q_3}{\mathbf{E}_3}.$$

The numerical results for q_3 and $\mathbf{E}_3$ confirm the above relationships. In Steps 2–4 the model is charged in different ways verifying the same piezoelectrical material parameters as in Step 1. In Step 2 the potentials of the bottom and the top surface are switched. In Step 3 a nodal concentrated electrical charge is applied, and in Step 4 a distributed electrical charge is applied instead of prescribing the potentials. In Step 5 a potential gradient is applied in the local 1-direction to verify the piezoelectric properties $d^{\varphi}_{1\,13}$, $g^{\varphi}_{1\,13}$, and $D^{\varphi(\sigma)}_{11}$.

In Steps 6–7 an open circuit condition is applied (the potential gradient is not prescribed by voltage boundary conditions), which results in reaction charges that are equal to zero. The piezoelectric constitutive equations can

be written in different forms. In particular, the strain can be expressed in terms of either the potential gradient $\mathbf{E}$ or the charge density $\mathbf{q}$. If the constitutive relation is expressed in terms of the potential gradient, the compliance data (typically denoted as $\mathbf{S}^{E}$ in the piezoelectric literature) define the mechanical behavior at zero potential gradient. In Abaqus the stiffness data at zero potential gradient are used to specify the mechanical behavior. If the constitutive relation is expressed using the charge density, the compliance matrix (typically denoted as $\mathbf{S}^{D}$ in the piezoelectric literature) defines the mechanical behavior at zero charge density. The compliance $\mathbf{S}^{D}$ can be obtained from the compliance $\mathbf{S}^{E}$ and the electrical properties. For the PZT-5H material, $S_{11}^{D} = 14.05 \times 10^{-12}$, $S_{12}^{D} = -7.27 \times 10^{-12}$, $S_{13}^{D} = -3.05 \times 10^{-12}$, $S_{31}^{D} = -3.05 \times 10^{-12}$, and $S_{33}^{D} = 8.99 \times 10^{-12}$. By loading the model at zero charge (in open circuit condition), these elastic compliances are verified.

Input files

ppzovallin.inp	Geometrically linear static analysis used to validate piezoelectric material properties.
ppzovalnlg.inp	Geometrically nonlinear static analysis used to validate piezoelectric material properties.
ppzovalnlg_tfv.inp	Geometrically nonlinear static analysis used to validate temperature- and field-variable-dependent piezoelectric material properties.

Submodeling

Submodeling is the technique whereby a portion of a structure is analyzed with a different (usually finer) mesh by "driving" the nodes on the boundary of that mesh from the interpolated solution on the original "global" mesh. To perform a submodel analysis, nodal quantities such as displacements, temperatures, pressures, displacement phases, etc. must be saved on the file output in the global analysis (usually done with a coarse mesh). The global model file output is attached to the submodel run (via the **globalmodel** parameter on the Abaqus execution procedure) to drive the boundary nodes on the submodel (usually done with a finer mesh). The same reference frame must be used in both models. The global and submodel meshes can have different element types within the same group of elements: planar solid to planar solid, axisymmetric solid to axisymmetric solid, three-dimensional solid to three-dimensional solid, general shell to general shell, etc. For shell-to-solid submodeling the global model consists of shell elements and the submodel consists of three-dimensional continuum elements. The procedure types can be different between the global analysis and the submodel analysis. For example, a linear static analysis in the global model can drive an elastic-plastic static solution in the submodel (as long as plasticity will not influence the driven boundary nodes), or a dynamic analysis in the global model can drive a static solution in a submodel (this assumes that inertia forces can be neglected at the submodel level). In addition, the global procedure can be performed in Abaqus/Standard to drive a submodeling procedure in Abaqus/Explicit and vice versa. For example, an Abaqus/Standard static analysis in the global model can drive a quasi-static Abaqus/Explicit analysis in the submodel.

The verification tests are divided into sections according to the element types supported in the submodel capability. Within each section a combination of elements and procedures is tested on small models with a limited number of elements. The values (or amplitudes) at the driven nodes, interpolated from the global analysis, are verified. In most cases the stress and strain fields in the submodel analysis match the results of the global analysis. However, in certain problems the meshes are too coarse to produce good agreement in stress and strain.

Each test consists of two input files: the global analysis and the submodel analysis. The same global file can drive several submodel analysis runs, each using a different mesh with elements that may or may not be the same as in the global analysis.

An example of running a sequentially coupled thermal-stress analysis is also given.

In this section:

- *General submodeling*
- *Two-dimensional continuum stress/displacement submodeling*
- *Three-dimensional continuum stress/displacement submodeling*
- *Cylindrical continuum stress/displacement submodeling*
- *Axisymmetric continuum stress/displacement submodeling*
- *Axisymmetric stress/displacement submodeling with twist*
- *Membrane submodeling*
- *Shell submodeling*
- *Surface element submodeling*
- *Heat transfer submodeling*
- *Coupled temperature-displacement submodeling*
- *Pore pressure submodeling*
- *Piezoelectric submodeling*
- *Acoustic submodeling*
- *Shell-to-solid submodeling*
- *Gasket submodeling*

- *Miscellaneous submodeling tests*

General submodeling

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

A general submodel uses beams or shells in the global model and shells or solids in the submodel.

Elements tested

B31 C3D8R C3D10HS C3D20RHT CAX6MP CPEG4 CPS4R DC2D4 S4R

Features tested

These problems test general submodeling for various procedures.

Problem description

The global model *GlobalBeam.inp* is a cantilever beam model that has a channel section. The submodel meshes the same channel section using either solid elements (*SubmodelSolid.inp*) or shell elements (*SubmodelShell.inp*). A similar analysis is done for a steady-state dynamic procedure with global model *GlobalBeamDynamic.inp* and submodel *SubmodelSolidDynamic.inp*. Submodel *SubmodelSolidPert.inp* uses a solid submodel that has a static perturbation procedure.

Global model *GlobalSkinStringer.inp* is a cantilever plate stiffened with beam stiffeners subjected to uniform pressure. The submodel *SkinStringerSubmodelBeamShell.inp* has a detailed model of a portion of the cantilever plate where the beam cross section is either meshed with solid elements or shell elements. The plate itself is modeled using shell elements in both the global model and the submodel.

Global model *GbStoS.inp* is a cantilever plate in which one half is meshed with solid elements and the other half is meshed with shell elements. The two halves are joined using shell-to-solid couplings. The submodel *SubStoS.inp* is made of solid elements only. One cut surface of the submodel lies over the solid elements of the global model, while the other cut surface lies over shell elements in the global model.

Global model *GlobalCPS4.inp* and submodel *SubmodelCPS4.inp* use plane stress elements in a static general procedure. Global modal *Global_C3D10.inp* and submodel *Submodel_C3D10.inp* use quadratic tetrahedral elements in a static general procedure. Submodel *Submodel_CPEG.inp* uses global model *pgcg4sfs.inp* to verify the usage of generalized plane strain elements.

Submodel *Submodel_2DHT.inp* and submodel *Submodel_3DCHT.inp* use global model *pgc24dfs.inp* and *pgc3kths.inp*, respectively, in heat transfer analysis and coupled temperature-displacement analysis. Global model *pgca6pkd.inp* and submodel *Submodel_CAX6P.inp* have a soils consolidation analysis that use quadratic axisymmetric elements.

Beam-to-beam submodel *SubmodelBeamToBeam.inp* uses global model *GlobalBeam.inp*.

Submodel *SubmodelSolidLcase.inp* is a static perturbation analysis that has load cases and makes use of *GlobalBeamLcase.inp*, which also has load cases. Global model *GlobalBeamDynamicLoadCase.inp* is a cantilever beam in a steady-state dynamic analysis with load cases, and the corresponding submodel *SubmodelSolidDynamicLoadCase.inp* is a steady-state dynamic analysis with load cases and solid elements.

Global model *GlobalBeamLcase.inp* and submodel *SubmodelSolidLcase.inp* verify the use of multiple linear load cases (*Multiple Linear Load Case Analysis*) in the context of submodeling. The global model uses beam

elements, and the submodel is meshed with solid elements. The global and submodel simulations each have a three-step sequence of a linear perturbation step with two transverse load cases, a general step with tensile loading and geometric nonlinearity considered, and then another linear perturbation step with same two transverse load cases as the first step. The first step involves a linear load case with transverse loading in the flange direction and another linear load case in the web direction. The third step has identical loading as the first step but has a different base state than the first step. Overlay plots of displacement solutions for each of the load cases in the first step and third step are shown in *Figure 18* and *Figure 19*.

Results and discussion

The results for submodel analysis are verified by comparing submodel solution with global model solution. Overlay plots of global model and submodel solution are shown in the figures below.

Input files

GlobalBeam.inp	Global beam model with general static procedure and nonlinear geometry.
GlobalBeamDynamic.inp	Global beam model with steady-state dynamic procedure.
SubmodelSolid.inp	Solid submodel with static procedure.
SubmodelShell.inp	Shell submodel file with static procedure.
SubmodelSolidDynamic.inp	Solid submodel with steady-state dynamic procedure.
SubmodelSolidPert.inp	Solid submodel with static perturbation procedure.
GlobalBeamElset.inp	Global beam model that defines element set to be used in submodel.
SubmodelSolidElset.inp	Solid submodel that uses global element set.
Global_BeamToShell.inp	Global beam model with general static procedure.
Submodel_BeamToShell.inp	Shell submodel with general static procedure.
GlobalSkinStringer.inp	Global stiffened plate model with general static procedure and nonlinear geometry.
SkinStringerSubmodelBeamShell.inp	Submodel of the stiffened plate with stiffeners.
GbStoS.inp	Global model of plate using solid and shell elements.
SubStoS.inp	Submodel using solid elements.
GlobalCPS4.inp	Global model using plane stress elements.
SubmodelCPS4.inp	Submodel using plane stress elements.
Global_C3D10.inp	Global model using C3D10 elements.
Submodel_C3D10.inp	Submodel using C3D10 elements.
pgcg4sfs.inp	Global model using generalized plane strain elements.
Submodel_CPEG.inp	Submodel using generalized plane strain elements.
pgc24dfs.inp	Global model for heat transfer analysis.
Submodel_2DHT.inp	Submodel for heat transfer analysis.
pgc3kths.inp	Global model with coupled temperature-displacement analysis.
Submodel_3DCHT.inp	Submodel with coupled temperature-displacement analysis.
pgca6pkd.inp	Global model with soils consolidation analysis.
Submodel_CAX6P.inp	Submodel with soils consolidation analysis.
SubmodelBeamToBeam.inp	Submodel with beam elements.
pgca3afd.inp	Global model with acoustic elements.
Submodel_ACAX3.inp	Submodel with acoustic elements.
GlobalBeamDynamicLoadCase.inp	Global model with steady-state dynamic procedure with load cases.
SubmodelSolidDynamicLoadCase.inp	Submodel with steady-state dynamic procedure with load cases.
GlobalBeamLcase.inp	Global beam model with beam elements and multiple linear load cases.
SubmodelSolidLcase.inp	Submodel with sold elements and multiple linear load cases.

Figures

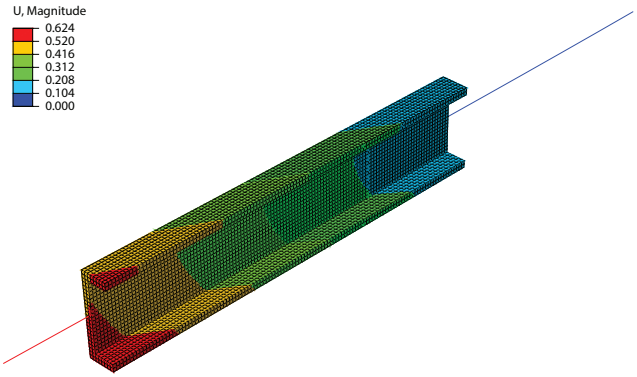


Figure 1: Overlay of global model and submodel for SubmodelSolid.

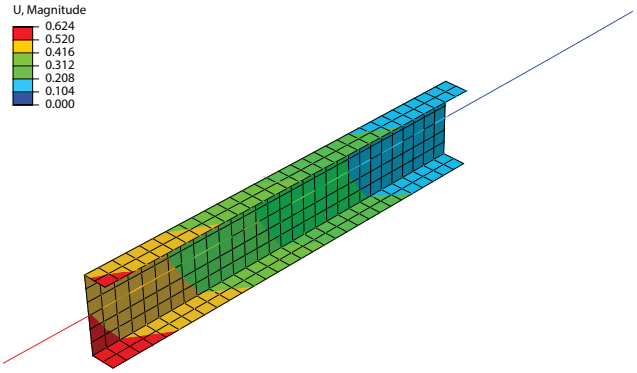


Figure 2: Overlay of global model and submodel for SubmodelShell.

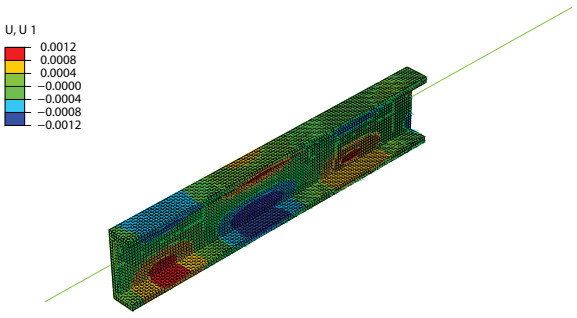


Figure 3: Overlay of global model and submodel for SubmodelSolidDynamic.

Lorem ipsum

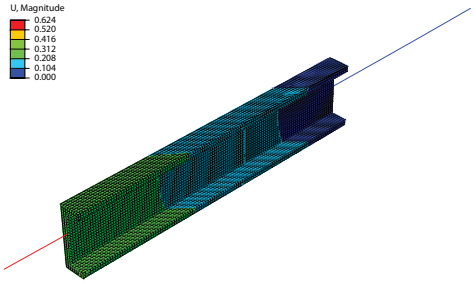


Figure 4: Overlay of global model and submodel for SubmodelSolidPert.

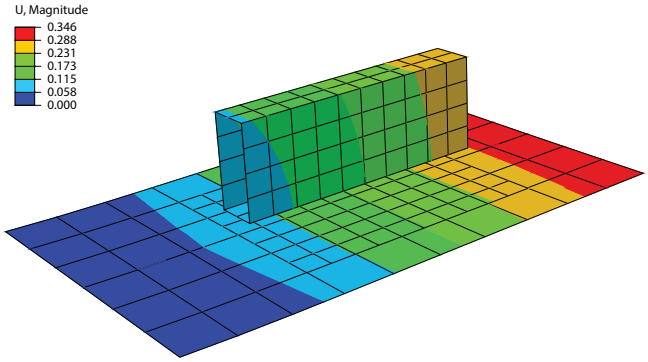


Figure 5: Overlay of global model and submodel for Submodel_BeamToShell.

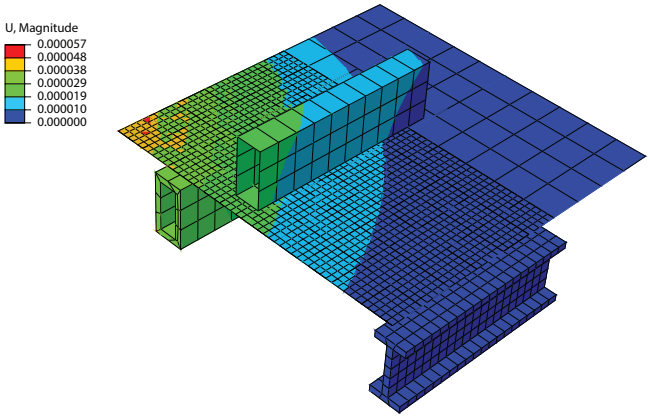


Figure 6: Overlay of global model and submodel for SkinStringerSubmodelBeamShell.

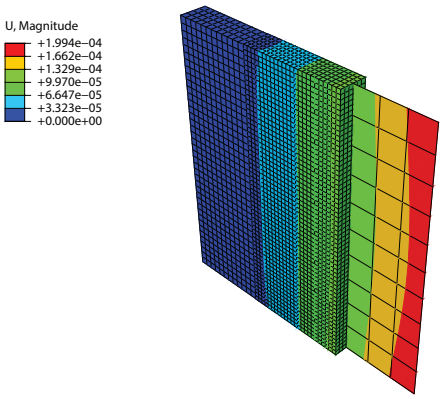


Figure 7: Overlay of global model and submodel for SubStoS.

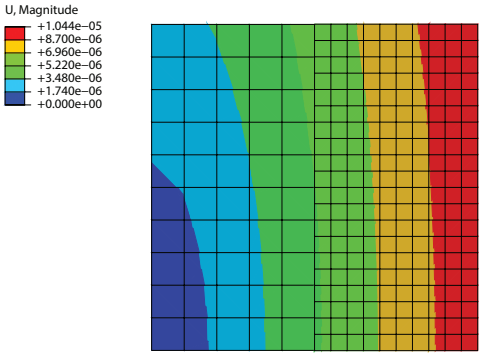


Figure 8: Overlay of global model and submodel for SubmodelCPS4.

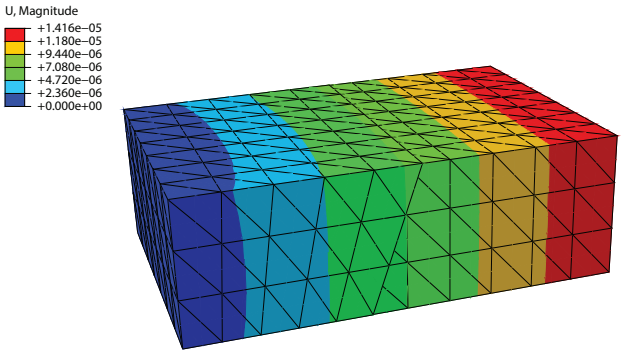


Figure 9: Overlay of global model and submodel for Submodel_C3D10.

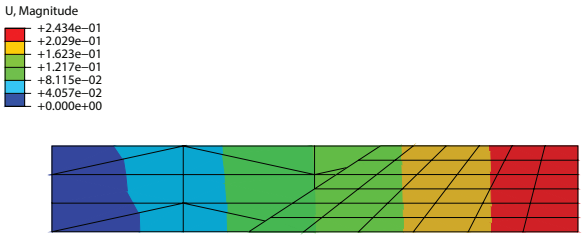


Figure 10: Overlay for global model and submodel for Submodel_CPEG.

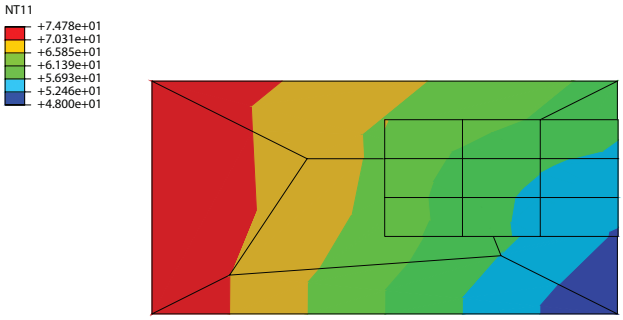


Figure 11: Overlay of global model and submodel for Submodel_2DHT.

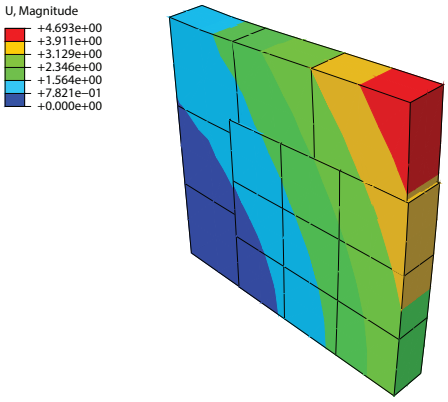


Figure 12: Overlay of global model and submodel for Submodel_3DCHT.

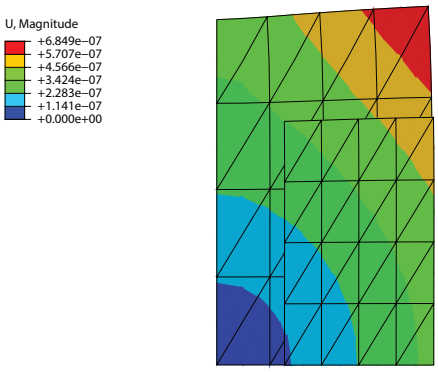


Figure 13: Overlay of global model and submodel for Submodel_CAX6P.

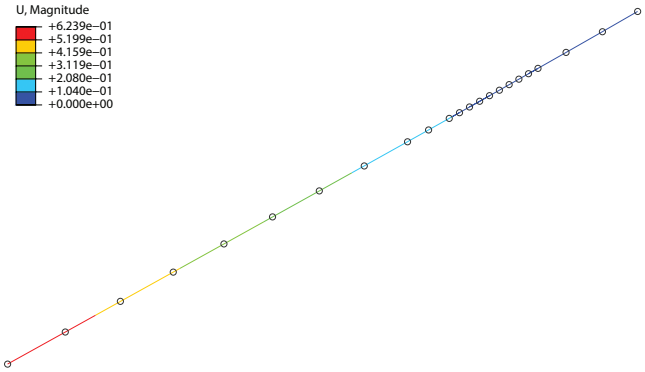


Figure 14: Overlay of global model and submodel for SubmodelBeamToBeam.

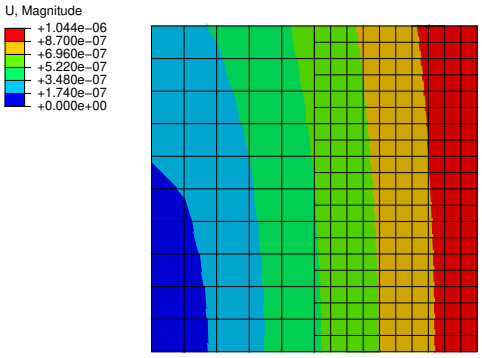


Figure 15: Overlay of global model and submodel for SubmodelSolidLcase.

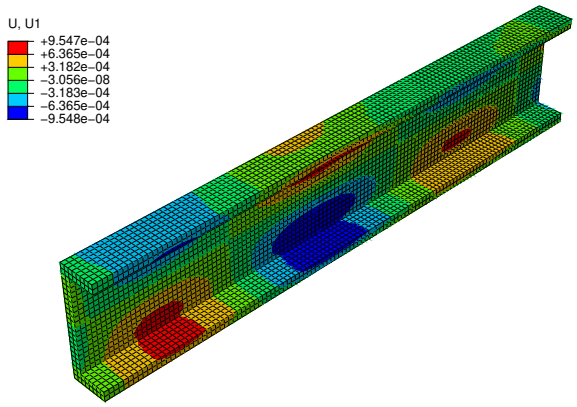


Figure 16: Submodel for SubmodelSolidDynamicLoadCase.

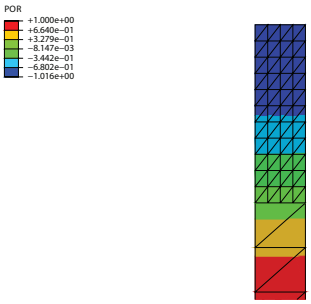


Figure 17: Overlay of global model and submodel for acoustic elements.

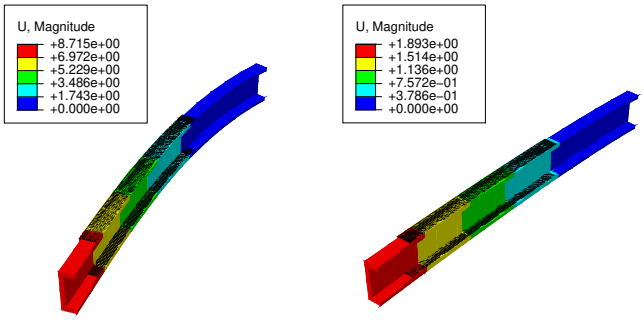


Figure 18: Overlay plot of global model and submodel displacement plot for linear load case 1 for step 1 (left) and step3 (right).

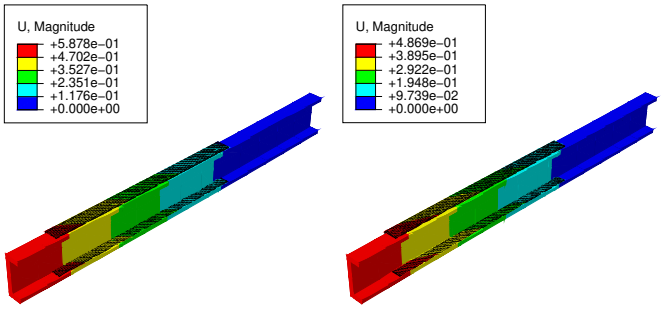


Figure 19: Overlay plot of global model and submodel displacement plot for linear load case 2 for step 1 (left) and step (3) right.

Two-dimensional continuum stress/displacement submodeling

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

CPEG3 CPEG4 CPEG6 CPEG6M CPEG8
CPE3 CPE4 CPE4H CPE4R CPE6 CPE6M CPE8 CPE8H CPE8R
CPS3 CPS4 CPS4R CPS6 CPS6M CPS8

Features tested

The submodeling capability is applied to two-dimensional continuum stress/displacement elements. In Abaqus/Standard general static and linear perturbation procedures are used in various combinations for both the global and submodel analyses. In Abaqus/Explicit the procedures are quasi-static for both the global and submodel analyses.

Problem description

Model: All global models have dimensions 8.0 × 1.5 in the *x*–*y* plane, with an out-of-plane dimension of 1.0 (plane stress analysis).

Material:

Young's modulus	3×10^6
Poisson's ratio	0.3
Density	10.0
Rayleigh damping (α_R)	0.2
Rayleigh damping (β_R)	0.4

Loading and boundary conditions: All global models involving static procedures and Abaqus/Explicit quasi-static procedures are subject to the loading and boundary conditions depicted in *Figure 1*. In Abaqus/Standard the time history of the loading, the time at which the corresponding submodeled analysis is performed, and the requested file output from the global model are unique to each individual analysis. In Abaqus/Explicit the same step time and smooth step loading are used in both the global and submodel analyses.

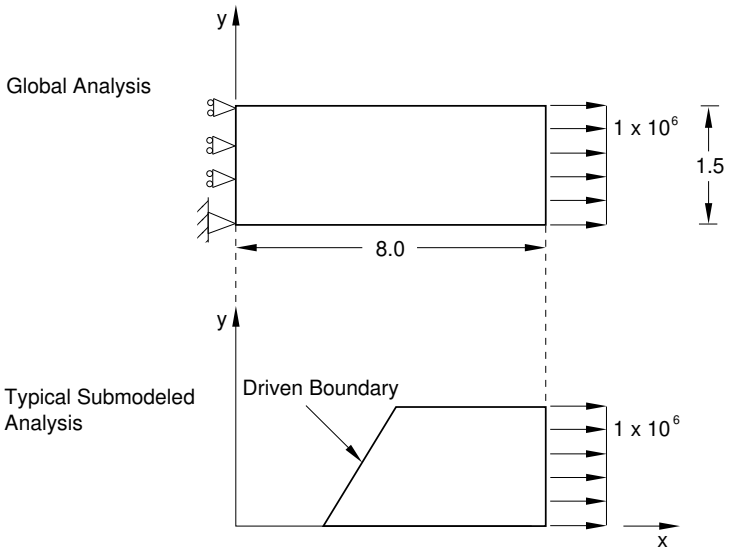


Figure 1: Global and submodeled domains used with static procedures.

All global models involving dynamic procedures in Abaqus/Standard are subject to the loading and boundary conditions depicted in *Figure 2*. For the transient simulations using the direct-integration implicit dynamic procedure, different excitation frequencies of the load can be tested by changing the parameters defined in the input files. As in the static analyses the time history of the loading, the time at which the corresponding submodeled analysis is performed, and the requested file output from the global model are unique to each individual analysis.

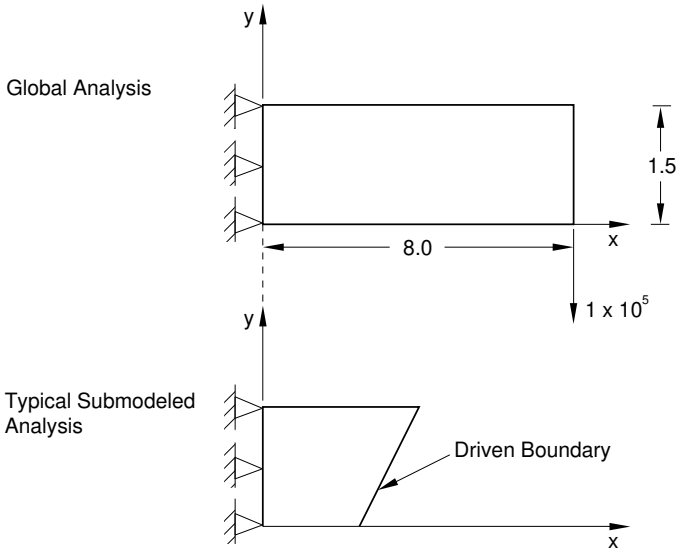


Figure 2: Global and submodeled domains used with dynamic procedures.

Results and discussion

The amplitudes of all driven variables in the submodeled analysis are correctly identified on the global analysis file output and applied at the driven nodes in the submodel analysis.

Input files

Abaqus/Standard input files

The following input files test various combinations of static analyses using the static and linear static perturbation procedures:

pgcg4sfs.inp	CPEG3, CPEG4 elements; global analysis.
pscg4sf1.inp	CPEG4 elements; submodel analysis of pgcg4sfs.inp.
pscg4sf1_sb.inp	CPEG4 elements; stress-based submodel analysis of pgcg4sfs.inp.
pscg4sf2.inp	Restart from pscg4sf1.inp.
pscg4sf2_sb.inp	Restart from pscg4sf1_sb.inp.
pgcg8sfs.inp	CPEG6, CPEG8 elements; global analysis.
pscg8sf1.inp	CPEG4 elements; submodel analysis of pgcg8sfs.inp.
pgcg8sks.inp	CPEG6M, CPEG8 elements; global analysis.
pscg8sk1.inp	CPEG4 elements; submodel analysis of pgcg8sks.inp.
pscg8sk1_sb.inp	CPEG4 elements; stress-based submodel analysis of pgcg8sks.inp.
pgce4sfs.inp	CPE3, CPE4 elements; global analysis.
psce4sf1.inp	CPE4 elements; submodel analysis of pgce4sfs.inp.
psce4sf1_sb.inp	CPE4 elements; stress-based submodel analysis of pgce4sfs.inp.
pgce4sfsg.inp	CPE3, CPE4 elements; **SUBMODEL*, GLOBAL ELSET; global analysis.
psce4sf1g.inp	CPE4 elements; **SUBMODEL*, GLOBAL ELSET; submodel analysis of pgce4sfsg.inp.
pgce4shm.inp	CPE4H elements; global analysis.
psce4sh1.inp	CPE4 elements; submodel analysis of pgce4shm.inp .
pgce4srm.inp	CPE4R elements; global analysis.
psce4sr1.inp	CPE4 elements; submodel analysis of pgce4srm.inp.
pgce8sfs.inp	CPE6, CPE8 elements; global analysis.
psce8sf1.inp	CPE4 elements; submodel analysis of pgce8sfs.inp.
pgce6sms.inp	CPE6M elements; global analysis.
psce6sm1.inp	CPE6M elements; submodel analysis of pgce6sms.inp.
psce6sm1_sb.inp	CPE6M elements; stress-based submodel analysis of pgce6sms.inp.
pgcs4sfs.inp	CPS3, CPS4 elements; global analysis.
pscs4sf1.inp	CPS4 elements; submodel analysis of pgcs4sfs.inp.
pgcs6sms.inp	CPS6M elements; global analysis.
pscs6sm1.inp	CPS6M elements; submodel analysis of pgcs6sms.inp.

The following input files test various combinations of dynamic analyses using the direct solution steady-state dynamic and direct-integration implicit procedures:

pgce8shd.inp	CPE8H elements; global analysis.
psce8sh1.inp	CPS8 elements; submodel analysis of pgce8shd.inp.
pgce8srd.inp	CPE8R elements; global analysis.
psce8sr1.inp	CPE8 elements; submodel analysis of pgce8srd.inp.
psce8sr1_sb.inp	CPE8 elements; stress-based submodel analysis of pgce8srd.inp.
pgcs8sfd.inp	CPS6, CPS8 elements; global analysis.

pscs8sf1.inp	CPS8 elements; submodel analysis of pgcs8sfd.inp.
pscs8sf1_sb.inp	CPS8 elements; stress-based submodel analysis of pgcs8sfd.inp.
submodel2delem_cpe8h_gd_std.inp	CPE8H elements; global **DYNAMIC* analysis.
submodel2delem_cps8_sd_std.inp	CPS8 elements; submodel **DYNAMIC* analysis.

Abaqus/Explicit input files

submodel2delem_cpe_g_xpl.inp	CPE4R, CPE3 elements; global analysis.
submodel2delem_cpe4r_s_xpl.inp	CPE4R elements; submodel analysis.
submodel2delem_cps_g_xpl.inp	CPS4R, CPS3 elements; global analysis.
submodel2delem_cps4r_s_xpl.inp	CPS4R elements; submodel analysis.
submodel2delem_cpe6m_g_xpl.inp	CPE6M elements; global analysis.
submodel2delem_cpe6m_s_xpl.inp	CPE6M elements; submodel analysis.
submodel2delem_cps6m_g_xpl.inp	CPS6M elements; global analysis.
submodel2delem_cps6m_s_xpl.inp	CPS6M elements; submodel analysis.

Three-dimensional continuum stress/displacement submodeling

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D4 C3D5 C3D6 C3D8 C3D8R C3D10 C3D10HS C3D10M C3D15 C3D20 C3D27 SC6R SC8R

Features tested

The submodeling capability is applied to the three-dimensional continuum stress/displacement elements and the continuum shell elements. In Abaqus/Standard general static, dynamic, and linear perturbation procedures are used in various combinations for both the global and submodel analyses. In Abaqus/Explicit both the global model and submodel are quasi-static analyses, except for the test that uses the GLOBAL ELSET parameter, which is a dynamic process. In the quasi-static tests each submodel can be driven from any of the global models.

The submodeling capability is also tested for a directly input matrix representing an element stiffness in static procedures (general and linear perturbation) in Abaqus/Standard. The scaling parameter in the submodel boundary condition is tested in an Abaqus/Explicit submodel analysis.

Problem description

Model: All global models have dimensions $8.0 \times 1.5 \times 1.0$.

Material:

Young's modulus	3×10^6
Poisson's ratio	0.3
Density	10.0
Rayleigh damping (α_R)	0.2
Rayleigh damping (β_R)	0.4

Loading and boundary conditions: All global models involving static procedures in Abaqus/Standard and all the analyses in Abaqus/Explicit are subject to the same loading and boundary conditions as depicted in *Two-dimensional continuum stress/displacement submodeling*. The time history of the loading, the time at which the corresponding submodeled analysis is performed, and the requested file output from the global model are unique to each individual analysis. Smooth-step amplitudes are used to load the quasi-static models in Abaqus/Explicit.

All global models involving dynamic procedures in Abaqus/Standard are subject to the same loading and boundary conditions as depicted in *Two-dimensional continuum stress/displacement submodeling*. For the transient simulations using the direct-integration implicit dynamic procedure, different excitation frequencies of the load can be tested by changing the parameters defined in the input files. As in the static analyses the time history of

the loading, the time at which the corresponding submodeled analysis is performed, and the requested file output from the global model are unique to each individual analysis.

Results and discussion

The amplitudes of all driven variables in the submodeled analysis are correctly identified in the global analysis file output and applied at the driven nodes in the submodel analysis.

Input files

Abaqus/Standard input files

The following input files test various combinations of static analyses using the static and linear static perturbation procedures:

pgc34sfs.inp	C3D4 elements; global analysis.
psc34sf1.inp	C3D4 elements; submodel analysis.
psc34sf1_sb.inp	C3D4 elements; stress-based submodel analysis.
pgc35sfs.inp	C3D5 elements; global analysis.
psc35sf1.inp	C3D5 elements; submodel analysis.
psc35sf1_sb.inp	C3D5 elements; stress-based submodel analysis.
pgc36sfs.inp	C3D6 elements; global analysis.
psc36sf1.inp	C3D6 elements; submodel analysis.
pgc38sfm.inp	C3D8 elements; global analysis.
psc38sf1.inp	C3D8 elements; **MATRIX INPUT* and **MATRIX ASSEMBLE*; submodel analysis.
psc38sf1_sb.inp	C3D8 elements; **MATRIX INPUT* and **MATRIX ASSEMBLE*; stress-based submodel analysis.
psc38sf1_stiff.inp	Matrix representing element stiffness for submodel analysis.
pgc38sfmg.inp	C3D8 elements; **SUBMODEL*, GLOBAL ELSET; global analysis.
psc38sf1g.inp	C3D8 elements; **SUBMODEL*, GLOBAL ELSET; submodel analysis.
pgc38sfmgm.inp	C3D8 elements; multiple **SUBMODEL* options; global analysis.
psc38sf1gm.inp	C3D8 elements; multiple **SUBMODEL* options; submodel analysis.
psc38sf1gm_sb.inp	C3D8 elements; multiple **SUBMODEL* options; stress-based submodel analysis.
pgc3tsfs.inp	C3D10 elements; global analysis.
pgc3tsfs_c3d10hs.inp	C3D10HS elements; global analysis.
psc3tsf1.inp	C3D10 elements; submodel analysis.
pgc3msfs.inp	C3D10M elements; global analysis.
psc3msf1.inp	C3D10M elements; submodel analysis.
psc3msf1_sb.inp	C3D10M elements; stress-based submodel analysis.
pgc3ksfs.inp	C3D20 elements; global analysis.
psc3ksf1.inp	C3D20 elements; submodel analysis.
psc3ksf1_sb.inp	C3D20 elements; stress-based submodel analysis.
pgc3rsfm.inp	C3D27 elements; global analysis.
psc3rsf1.inp	C3D27 elements; submodel analysis.

psc3rsf1_sb.inp	C3D27 elements; stress-based submodel analysis.
pgc3rsfmg.inp	C3D27 elements; **SUBMODEL*, GLOBAL ELSET; global analysis.
psc3rsf1g.inp	C3D27 elements; **SUBMODEL*, GLOBAL ELSET; submodel analysis.
psc3rsf1g_sb.inp	C3D27 elements; **SUBMODEL*, GLOBAL ELSET; stress-based submodel analysis.

The following input files test the submodeling capability using the dynamic procedures:

pgc3fsfd.inp	C3D15 elements; global **MODAL DYNAMIC* analysis.
psc3fsf1.inp	C3D15 elements; submodel **MODAL DYNAMIC* analysis.
psc3fsf1_sb.inp	C3D15 elements; stress-based submodel analysis.
submodel3delem_c3d15_gd_std.inp	C3D15 elements; global **DYNAMIC* analysis.
submodel3delem_c3d8_sd_std.inp	C3D8 elements; submodel **DYNAMIC* analysis.

Abaqus/Explicit input files

submodel3delem_c3d4_g_xpl.inp	C3D4 elements; global analysis.
submodel3delem_c3d6_g_xpl.inp	C3D6 elements; global analysis.
submodel3delem_c3d8_g_xpl.inp	C3D8 elements; global analysis.
submodel3delem_c3d8r_g_xpl.inp	C3D8R elements; global analysis.
submodel3delem_c3d10m_g_xpl.inp	C3D10M elements; global analysis.
submodel3delem_c3d8_s_xpl.inp	C3D8 elements; submodel analysis.
submodel3delem_c3d8r_s_xpl.inp	C3D8R elements; **BOUNDARY*, SUBMODEL, SCALE; submodel analysis.
submodel3delem_c3d10m_s_xpl.inp	C3D10M elements; submodel analysis.
submodel3delem_sc6r_g_xpl.inp	SC6R elements; global analysis.
submodel3delem_sc8r_g_xpl.inp	SC8R elements; global analysis.
submodel3delem_sc8r_s_xpl.inp	SC8R elements; submodel analysis.
submodel3delem_c3d8r_g_gel_x.inp	C3D8R elements; **SUBMODEL*, GLOBAL ELSET; global analysis.
submodel3delem_c3d8r_s_gel_x.inp	C3D8R elements; **SUBMODEL*, GLOBAL ELSET; submodel analysis.
submodel3delem_sc8r_g_gel_x.inp	SC8R elements; **SUBMODEL*, GLOBAL ELSET; global analysis.
submodel3delem_sc8r_s_gel_x.inp	SC8R elements; **SUBMODEL*, GLOBAL ELSET; submodel analysis.

Cylindrical continuum stress/displacement submodeling

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

CCL9 CCL12 CCL18 CCL24

Features tested

The submodeling capability is applied to cylindrical elements. The general static procedure is used in various combinations for both the global and submodel analyses.

Problem description

Model: All global models consist of an 180° cylindrical segment with an outer radius of 2 and an inner radius of 1. The submodel input files model the right half of the global model. The submodel driven nodes lie along the symmetrical plane of the global model.

Material:

Young's modulus	6×10^6
Poisson's ratio	0

Loading and boundary conditions: All the nodes at both ends are fixed in the circumferential direction, and all the nodes at the inner surface are constrained in all degrees of freedom. Distributed loads or prescribed displacements are applied at the outer surface of the cylindrical elements.

Results and discussion

The amplitudes of all driven variables in the submodeled analysis are correctly identified on the global analysis file output and applied at the driven nodes in the submodel analysis.

Input files

Abaqus/Standard input files

xembedele3d10_std.inp	CCL12 elements; global analysis.
xembedele3d10_submodel_std.inp	CCL12 elements; submodel analysis.
xembedele3d11_std.inp	CCL24 elements; global analysis.
xembedele3d11_submodel_std.inp	CCL24 elements; submodel analysis.
xembedele3d12_std.inp	CCL9 elements; global analysis.
xembedele3d12_submodel_std.inp	CCL9 elements; submodel analysis.

xembedele3d13_std.inp	CCL18 elements; global analysis.
xembedele3d13_submodel_std.inp	CCL18 elements; submodel analysis.

Axisymmetric continuum stress/displacement submodeling

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

CAX3 CAX4I CAX4R CAX6 CAX6M CAX8H CAX8R

Features tested

The submodeling capability is applied to axisymmetric stress/displacement elements. The submodels cut the global model along lines that are diagonal to the global model's regular geometry. Static and steady-state dynamic procedures are tested in Abaqus/Standard. In Abaqus/Explicit the analyses are quasi-static; each submodel can be driven from any of the global models.

Problem description

Model: The global model dimensions are 8.0×1.5 in the r–z plane.

Material:

Young's modulus	30×10^6
Poisson's ratio	0.3
Density	10.0

Boundary conditions: Along the left face of the model $u_r=0$. One node is further constrained so that $u_z=0$.

Loading: In all models a pressure of -1.0×10^6 is applied to the right-hand face. (A figure with the same geometry and loading is shown in *Two-dimensional continuum stress/displacement submodeling*, except that the figure has rectangular axes x and y instead of the axisymmetric axes r and z.)

Results and discussion

In all cases the nodal displacements for the driven nodes in the submodels are correctly interpolated from the global model results. In the cases presented, element and nodal responses in the submodels match the responses in the global models.

Input files

Abaqus/Standard input files

pgca4sis.inp	CAX4I elements; global analysis.
psca8si1.inp	CAX4I elements; submodel analysis.

psca8si1_sb.inp	CAX4I elements; stress-based submodel analysis.
pgca6sts.inp	CAX6 elements; global analysis.
psca6st1.inp	CAX6 elements; submodel analysis.
psca6st1_sb.inp	CAX6 elements; stress-based submodel analysis.
pgca6sms.inp	CAX6M elements; global analysis.
psca6sm1.inp	CAX6M elements; submodel analysis.
pgca8shs.inp	CAX8H elements; global analysis.
psca8sh1.inp	CAX8H elements; submodel analysis.
pgca8srm.inp	CAX8R elements; global analysis.
psca8sr1.inp	CAX8R elements; submodel analysis.
psca8sr1_sb.inp	CAX8R elements; stress-based submodel analysis.

Abaqus/Explicit input files

submodelaxielem_cax3_g_xpl.inp	CAX3 elements; global analysis.
submodelaxielem_cax3_s_xpl.inp	CAX3 elements; submodel analysis.
submodelaxielem_cax4r_g_xpl.inp	CAX4R elements; global analysis.
submodelaxielem_cax4r_s_xpl.inp	CAX4R elements; submodel analysis.
submodelaxielem_cax6m_g_xpl.inp	CAX6M elements; global analysis.
submodelaxielem_cax6m_s_xpl.inp	CAX6M elements; submodel analysis.

Axisymmetric stress/displacement submodeling with twist

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

CGAX4R CGAX4 CGAX6M CGAX6MH CGAX8H

Features tested

The submodeling capability is applied to axisymmetric stress/displacement elements with twist. The submodels cut the global model along lines that are diagonal to the global model's regular geometry. The static analysis procedure is used.

Problem description

Model: The global model dimensions are 8.0×1.5 in the r–z plane.

Material:

Young's modulus	30×10^6
Poisson's ratio	0.3
Density	1.0

Step I boundary conditions: Along the left face of the model u_r=0. One node is further constrained so that u_z=0.

Step I loading: In all models a pressure of -1.0×10^6 is applied to the right-hand face. (A figure with the same geometry and loading is shown in *Two-dimensional continuum stress/displacement submodeling*, except with rectangular Cartesian coordinate axes x and y instead of the cylindrical coordinate axes r and z.)

Step II boundary conditions: The bottom face is constrained in all degrees of freedom.

Step II loading: In all models a twist of 0.01 radians is applied to the top face.

Results and discussion

In all cases the nodal displacements for the driven nodes in the submodels are correctly interpolated from the global model results. In the cases presented, element and nodal responses in the submodels match the responses in the global models.

Input files

pgca4grs.inp	CGAX4R elements; global analysis.

psca8gf1.inp	CGAX4R elements; submodel analysis.
psca8gf1_sb.inp	CGAX4R elements; stress-based submodel analysis.
pgca4grsgm.inp	CGAX4R elements; multiple **SUBMODEL* options; global analysis.
psca8gf1gm.inp	CGAX4R elements; multiple **SUBMODEL* options; submodel analysis.
pgca4gfs.inp	CGAX4 elements; global analysis.
psca4gf1.inp	CGAX4 elements; submodel analysis.
pgca6gfs.inp	CGAX6M elements; global analysis.
psca6gf1.inp	CGAX6M elements; submodel analysis.
pgca6ghs.inp	CGAX6MH elements; global analysis.
psca6gh1.inp	CGAX6MH elements; submodel analysis.
pgca8ghs.inp	CGAX8H elements; global analysis.
psca8gh1.inp	CGAX8H elements; submodel analysis.

Membrane submodeling

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

M3D3 M3D4 M3D4R M3D6 M3D8R

Features tested

The submodeling capability is applied to membrane models. The global input files use the models from the membrane patch tests described in *Membrane patch test*. The submodel input files model the right half of the global model; in Abaqus/Standard the submodel has 16 M3D8 elements, and in Abaqus/Explicit the submodel uses 16 M3D4R or 32 M3D3 elements. The submodel driven nodes lie along the line parallel to the *y*-axis of the model. In Abaqus/Standard static perturbation and static general procedures are tested. In Abaqus/Explicit the analysis is performed as a quasi-static procedure; a velocity boundary condition that gives rise to the perturbation is specified instead of the perturbation step.

Problem description

The global models' dimensions and material properties are the same as in the patch tests used in *Membrane patch test*. The nodal file requests have been changed to enable the interpolation for the driven variables' values or driven nodes' history amplitudes. The submodel material properties are the same as the global model properties.

Results and discussion

All driven variables are correctly interpolated from the global analysis. Since the prescribed displacement and/or concentrated force patterns are brought to their physical positions on the submodel, the stress fields do not match in both models.

Input files

Abaqus/Standard input files

pgm33sfs.inp	M3D3 elements; global analysis.
psm38sf1.inp	M3D3 elements; submodel analysis.
pgm33sfsgm.inp	M3D3 elements; multiple **SUBMODEL* options; global analysis.
psm38sf1gm.inp	M3D3 elements; multiple **SUBMODEL* options; submodel analysis.
pgm34sfs.inp	M3D4 elements; global analysis.
psm38sf2.inp	M3D4 elements; submodel analysis.
pgm36sfs.inp	M3D6 elements; global analysis.
psm38sf3.inp	M3D6 elements; submodel analysis.
pgm38srs.inp	M3D8R elements; global analysis.
psm38sf4.inp	M3D8R elements; submodel analysis.

Abaqus/Explicit input files

submodelmemb_g_m3d3_xpl.inp	M3D3 elements; global analysis.
submodelmemb_s_m3d3_xpl.inp	M3D3 elements; submodel analysis.
submodelmemb_g_m3d4r_xpl.inp	M3D4R elements; global analysis.
submodelmemb_s_m3d4r_xpl.inp	M3D4R elements; submodel analysis.

Shell submodeling

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Bending tests

Elements tested

S3 S3R S3RS S4 S4R S4RS S4RSW S8R STRI3

Features tested

The submodeling capability is applied to various shell elements, with 6 degrees of freedom per node, subject to a bending load. Various combinations for both the global and submodel analyses are tested: in Abaqus/Standard general static and static perturbation procedures are used, and in Abaqus/Explicit the analyses are dynamic and quasi-static.

Problem description

Model: All global models have dimensions 10.0×3.0 in the x–y plane and use five section points through the thickness of 0.001.

Material:

Young's modulus	1×10^6
Poisson's ratio	0.3
Density	10

Loading and boundary conditions: Except for the problem defined in files *pgsf4srsgm.inp* and *pssf4sr1gm.inp*, the global model is constrained such that all displacement and rotation degrees of freedom for nodes along the y-axis are suppressed. All elements in the model are then subject to a uniform pressure load of 1×10^{-7} in the positive z-direction. In Abaqus/Explicit the elements are subject to a uniform pressure load of 1×10^{-2} in the positive z-direction. The global models using triangular shells in Abaqus/Explicit have three steps; however, the submodel analyses have one step that is driven from the third global step. This is valid because the inertial forces are not significant during the first two steps (the process is quasi-static).

The model considered in Abaqus/Standard files *pgsf4srsgm.inp* and *pssf4sr1gm.inp* and in Abaqus/Explicit input files using quadrilateral shells has two shell elements through the thickness in part of the region. One end of the model is fixed, while displacements in the z-direction are applied to the other end: in the positive z-direction for one layer of shells and in the opposite direction for the other layer. This is a special situation, which, in general, necessitates the use of multiple submodels to ensure that driven nodes are assigned to the correct global elements.

General: Gauss integration is used for the shell cross-section in input files *pgsf3srm.inp* and *pssf3sr1.inp*.

Results and discussion

The amplitudes of all driven variables in the submodel analysis are correctly identified in the global analysis file output and applied at the driven nodes in the submodel analysis.

Input files

Abaqus/Standard input files

pgsf3srm.inp	S3/S3R elements; global analysis.
pssf3sr1.inp	S3/S3R elements; submodel analysis.
pgsf3srmg.inp	S3/S3R elements; **SUBMODEL*, GLOBAL ELSET; global analysis.
pssf3sr1g.inp	S3/S3R elements; **SUBMODEL*, GLOBAL ELSET; submodel analysis.
pgse4sfs.inp	S4 elements; global analysis.
psse4sf5.inp	S4 elements; submodel analysis.
pgsf4srs.inp	S4R elements; global analysis.
pssf4sr1.inp	S4R elements; submodel analysis.
pgsf4srsgm.inp	S4R elements; multiple **SUBMODEL* options; global analysis.
pssf4sr1gm.inp	S4R elements; multiple **SUBMODEL* options; submodel analysis.
pgs68srm.inp	S8R elements; global analysis.
pss68sr1.inp	S8R elements; submodel analysis.
pgs63sfs.inp	STRI3 elements; global analysis.
pss63sf1.inp	STRI3 elements; submodel analysis.

Abaqus/Explicit input files

submodelshell_g_gel_s3r_xpl.inp	S3R elements; **SUBMODEL*, GLOBAL ELSET; global analysis.
submodelshell_s_gel_s3r_xpl.inp	S3R elements; **SUBMODEL*, GLOBAL ELSET; submodel analysis.
submodelshell_g_s3r_xpl.inp	S3R elements; global analysis.
submodelshell_s_s3r_xpl.inp	S3R elements; submodel analysis.
submodelshell_g_s3rs_xpl.inp	S3RS elements; global analysis.
submodelshell_s_s3rs_xpl.inp	S3RS elements; submodel analysis.
submodelshell_g_s4_xpl.inp	S4 elements; global analysis.
submodelshell_s_s4_xpl.inp	S4 elements; submodel analysis.
submodelshell_g_s4r_xpl.inp	S4R elements; global analysis.
submodelshell_s_s4r_xpl.inp	S4R elements; submodel analysis.
submodelshell_g_m_s4r_xpl.inp	S4R elements; multiple **SUBMODEL* options; global analysis.
submodelshell_s_m_s4r_xpl.inp	S4R elements; multiple **SUBMODEL* options; submodel analysis.
submodelshell_g_m_s4rs_xpl.inp	S4RS elements; multiple **SUBMODEL* options; global analysis.
submodelshell_s_m_s4rs_xpl.inp	S4RS elements; multiple **SUBMODEL* options; submodel analysis.

submodelshell_g_m_s4rsw_xpl.inp	S4RSW elements; multiple **SUBMODEL* options; global analysis.
submodelshell_s_m_s4rsw_xpl.inp	S4RSW elements; multiple **SUBMODEL* options; submodel analysis.

Membrane tests

Elements tested
S4R5

Features tested

The submodeling capability is applied to two patches of shell elements, with 5 degrees of freedom per node, subject to membrane-type loading. General static and static perturbation procedures are used in various combinations for both the global and submodel analyses.

Problem description

Model: The global models have dimensions 0.24×0.12 in the x–y plane and use five section points through the thickness of 0.001.

Material:

Young's modulus	1×10^6
Poisson's ratio	0.25

Loading and boundary conditions: $u_x = 10^{-3}(x + y/2)$, $u_y = 10^{-3}(y + x/2)$, at all exterior nodes. $u_z = 0$ at all nodes.

Results and discussion

The amplitudes of all driven variables (translational degrees of freedom in this case) in the submodel analysis are correctly identified in the file output for the global analysis and applied at the driven nodes in the submodel analysis.

Input files

pgs54srs.inp	S4R5 elements; global analysis.
pss54sr1.inp	S4R5 elements; submodel analysis.
pgs54srsg.inp	S4R5 elements; **SUBMODEL*, GLOBAL ELSET; global analysis.
pss54sr1g.inp	S4R5 elements; **SUBMODEL*, GLOBAL ELSET; submodel analysis.

Heat transfer test

Elements tested
DS3 DS6 DS8

Features tested

The submodeling capability is applied to a mesh of shell elements in a heat transfer analysis.

Problem description

Model: The global model has dimensions 10.0×3.0 in the x–y plane and uses three section points through the thickness of 0.001.

Material:

Thermal conductivity	1.0

Loading and boundary conditions: T=0.0 along x=y=0; and T=100.0 along x=10.0, y=3.0.

Results and discussion

The amplitudes of temperature in the submodel analysis are correctly identified in the global analysis file output and applied at the driven nodes in the submodel analysis.

Input files

pgs33dfh.inp	DS3 elements; global analysis.
pss33df1.inp	DS3 elements; submodel analysis.
pgs36dfh.inp	DS6 elements; global analysis.
pss36df1.inp	DS6 elements; submodel analysis.
pgs38dfh.inp	DS8 elements; global analysis.
pss38df1.inp	DS8 elements; submodel analysis.

Thermal-stress analysis

Elements tested

DS4 S4 S4R

Features tested

A sequentially coupled thermal-stress analysis using the submodeling technique is tested.

Problem description

Model: The global model has dimensions 3.0×2.0 in the x–z plane and uses three section points through the thickness of 0.001.

Material:

Young's modulus	1.0×10^{6}
Poisson's ratio	0.3
Thermal conductivity	4.85×10^{-4}
Coefficient of thermal expansion (α)	1.0×10^{-6}

Loading and boundary conditions: In the global heat transfer analysis a linear through-thickness temperature gradient is developed in the model by specifying T=0 at all nodes on the top face of the plate and

T=100 at all nodes on the bottom face. The global model for the thermal-stress analysis is constrained such that u_x=0 for x=0, u_y=0 for x=0 and x=3, and u_z=0 for x=y=z=0.

Results and discussion

Submodeling of a sequentially coupled thermal-stress analysis can be accomplished by any one of three methods in Abaqus. Whenever interpolation of temperature as a field variable is required between models because of mesh dissimilarities, temperatures must be read from the output database, since temperature interpolation is not supported with the results file. Driven variables can be interpolated using either the results file or the output database.

Method 1

1. Run the heat transfer analysis on the global model, and output the nodal temperatures.
2. Run the thermal-stress analysis on the global model, reading (and possibly interpolating) temperatures as field variables from the previous global heat transfer analysis. Output the nodal temperatures and displacements.
3. Run the submodel analysis reading (and possibly interpolating) temperatures as field variables and displacements from the global thermal-stress analysis.

Method 2

1. Run the heat transfer analysis on the global model, and output the nodal temperatures.
2. Run the thermal-stress analysis on the global model, reading (and possibly interpolating) temperatures as field variables from the previous global heat transfer analysis. Output the nodal temperatures and displacements.
3. Run the thermal-stress submodel analysis, reading (and possibly interpolating) temperatures as field variables from the global heat transfer analysis and displacements from the global thermal-stress analysis.

Method 3

1. Run the heat transfer analysis on the global model, and output the nodal temperatures.
2. Run a heat transfer submodel analysis, reading temperatures as driven from the global model. Output the nodal temperatures.
3. Run the thermal-stress submodel analysis, reading (and possibly interpolating) temperatures as field variables from the previous heat transfer submodel analysis.

The first two methods make use of the dissimilar mesh interpolation technique.

The amplitudes of all driven variables in the submodel analysis are correctly identified in the global analysis and applied at the driven nodes in the submodel analysis.

Input files

pgs34dfq.inp	DS4 elements; global heat transfer analysis.
pss34df1.inp	DS4 elements; submodel heat transfer analysis.
pgse4sfsc.inp	S4 elements; global static thermal-stress analysis.
psse4sf5.inp	S4 elements; submodel static thermal-stress analysis.

pgsf4srq.inp	S4R elements; global static thermal-stress analysis.
pssf4sr2.inp	S4R elements; submodel static thermal-stress analysis.
pssf4sr2_inter1.inp	Submodel thermal-stress analysis that interpolates temperatures from the global heat transfer analysis.
pssf4sr2_inter2.inp	Submodel thermal-stress analysis that interpolates temperatures from the global thermal-stress analysis.
pssf4sr2_2odb_inter.inp	Submodel thermal-stress analysis that interpolates temperatures from two different output database files representing heat transfer analyses.

Finite rotation test

Elements tested

S4R

Features tested

The submodeling capability is applied to a shell element, with 6 degrees of freedom per node, subjected to rotation boundary conditions in a large-displacement analysis. In Abaqus/Standard general static procedures are used for both the global and submodel analyses. In Abaqus/Explicit dynamic procedures are used for both analyses.

Problem description

Model: Both the global model and the submodel use a single element with dimensions 10.0×3.0 in the x–y plane, with a thickness of 0.001.

Material:

Young's modulus	1×10^6
Poisson's ratio	0.3
Density	10

Boundary conditions: The global model is constrained such that all displacement and rotation degrees of freedom for nodes along the y-axis are suppressed. The rotation degrees of freedom at the remaining nodes are given finite rotation boundary conditions in all three rotation components using different amplitude functions.

Results and discussion

The amplitudes of all driven variables in the submodeled analysis are correctly identified in the global analysis file output and applied at the driven nodes in the submodel analysis.

Input files

Abaqus/Standard input files

pgsf4srr.inp	S4R elements; global analysis.
pssf4sr3.inp	S4R elements; submodel analysis.

Abaqus/Explicit input files

submodelshell_grot_s4r_xpl.inp	S4R elements; global analysis.

submodelshell_srot_s4r_xpl.inp	S4R elements; submodel analysis.

Continuum shell elements

Elements tested

C3D8I SC6R SC8R S4 CSS8

Features tested

The submodeling capability is tested for continuum shell elements. The general static procedure is used for the global model as well as the submodel.

Problem description

In all the problems the global model is a cantilever beam loaded by concentrated loads at one end and fixed at the other end. The submodel consists of a partial cantilever beam that includes the fixed end.

Results and discussion

The amplitudes of all the driven variables in the submodel analysis are correctly identified in the global analysis output database and applied at the driven nodes in the submodel analysis.

Input files

global_sc8r_c3d8i.inp	SC8R elements; global analysis.
sub_sc8r_c3d8i.inp	C3D8I elements; submodel analysis.
global_sc6r_c3d8i.inp	SC6R elements; global analysis.
sub_sc6r_c3d8i.inp	C3D8I elements; submodel analysis.
global_shell_sc8r.inp	S4 elements; global analysis.
sub_shell_sc8r.inp	SC8R elements; submodel analysis.
global_sc8r_css8.inp	CSS8 elements; global analysis.
sub_sc8r_css8.inp	CSS8 elements; submodel analysis.

Surface element submodeling

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

SFM3D3 SFM3D4 SFM3D6 SFM3D8R

Features tested

The submodeling capability is applied to models with surface elements. The global input files use the models from the membrane patch tests described in *Membrane patch test*. The submodel input files model the right half of the global model with 16 SFM3D8 elements. The submodel driven nodes lie along the line parallel to the *y*-axis of the model. Static perturbation and static general procedures are tested.

Problem description

The global models' dimensions are the same as in the patch tests used in *Membrane patch test*. The nodal file requests have been changed to enable the interpolation for the driven variables' values or driven nodes' history amplitudes. There are three layers of rebar, which are oriented at 0°, 45°, and 90° from the *x*-axis, The material properties and dimensions of the rebar are as follows:

Young's modulus	1×10^6
Poisson's ratio	0.25
Rebar area	0.3
Rebar spacing	0.5

The submodel material properties are the same as the global model properties.

Results and discussion

All driven variables are interpolated correctly from the global analysis. Since the prescribed displacement and/or concentrated force patterns are brought to their physical positions on the submodel, the stress fields do not match in both models.

Input files

Abaqus/Standard input files

pgx33sfs_std.inp	SFM3D3 elements; global analysis.
psx38sf1_std.inp	SFM3D8 elements; submodel analysis.
pgx34sfs_std.inp	SFM3D4 elements; global analysis.
psx38sf2_std.inp	SFM3D8 elements; submodel analysis.
pgx36sfs_std.inp	SFM3D6 elements; global analysis.
psx38sf3_std.inp	SFM3D8 elements; submodel analysis.

pgx38srs_std.inp	SFM3D8R elements; global analysis.
psx38sf4_std.inp	SFM3D8 elements; submodel analysis.

Heat transfer submodeling

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Features tested

The submodeling capability is applied to heat transfer elements in Abaqus/Standard and to coupled temperature-displacement elements in Abaqus/Explicit. The thermal expansion coefficient is set to zero and dummy mechanical properties are used in Abaqus/Explicit analyses since only the thermal response is of interest. Three of the global model meshes are taken from element patch tests, while the fourth is a regular mesh. In Abaqus/Standard both steady-state and transient procedures are tested, while in Abaqus/Explicit the dynamic temperature-displacement procedure is used for all simulations.

DC2D4, CPE4RT, CPS4RT elements

Problem description

Model: The geometry is taken from the patch test in *ec24dfp4.inp*.

The global model dimensions are 0.24×0.12 in the x–y plane with a thickness of 1.0. One side of the submodel lies along the right-hand side of the global model, while the other three sides of the submodel lie completely inside the global model.

Material:

Conductivity	4.85×10^{-4}
Density	0.283
Specific heat	0.116

Loading: A uniform film with a reference sink temperature of 75 and a film coefficient of 0.103 is applied along the left edge of the global model. Nodal temperatures of 48 and 60 are applied to the lower right and upper right nodes of the global model, respectively.

Results and discussion

Transient heat transfer analysis is performed. The nodal temperatures for the driven nodes in the submodel are correctly interpolated from the global model results.

Input files

Abaqus/Standard input files

pgc24dfs.inp	Global analysis.
psc24df1.inp	Submodel analysis.

Abaqus/Explicit input files

submodelht_g_cpe4rt_xpl.inp	CPE4RT, global analysis.
submodelht_g_cps4rt_xpl.inp	CPS4RT, global analysis.
submodelht_s_cpe4rt_xpl.inp	CPE4RT, submodel analysis.
submodelht_s_cps4rt_xpl.inp	CPE4RT, submodel analysis.

DC2D6, CPE6MT, CPS6MT elements

Problem description

Model: The global model dimensions are 4×3 in the x–y plane with a thickness of 1.0. The submodel occupies the upper right-hand corner of the global model.

Material:

Conductivity	4.85×10^{-4}
Density	0.283
Specific heat	0.116

Loading: A body flux of 0.3 is applied on the entire global model and submodel. Nodal temperatures of 200 and 400 are prescribed along the left edge and the bottom edge of the global model, respectively.

Results and discussion

In Abaqus/Standard steady-state heat transfer analysis is performed. In Abaqus/Explicit a transient analysis is performed over a period of time in which the steady-state regime is reached. The nodal temperatures for the driven nodes in the submodel are correctly interpolated from the global model results. In *Figure 1* a temperature contour plot for the submodel overlays a contour plot of the global model. The temperature contours match at the boundaries between the global and submodels, showing that the driven nodal temperatures on the submodel are correct.

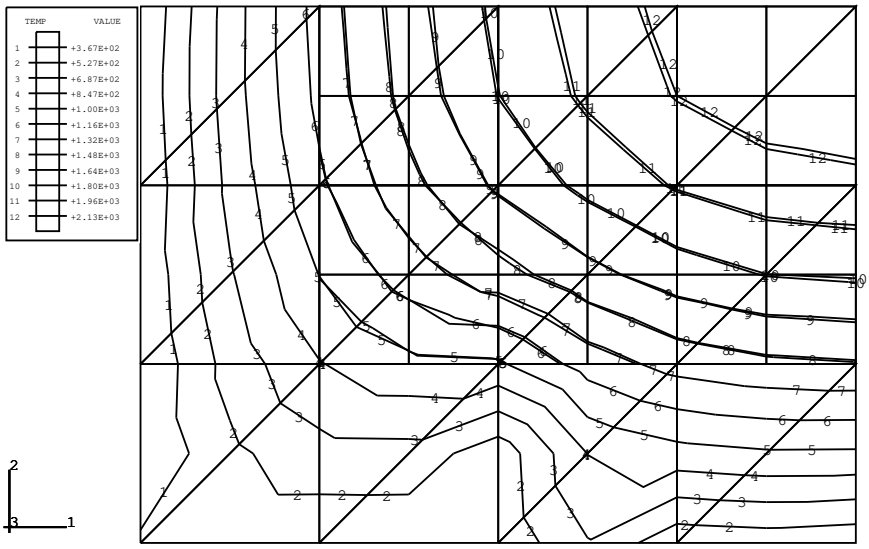


Figure 1: Temperature contours in global model and submodel with DC2D6 elements.

Input files

Abaqus/Standard input files

pgc26dfs.inp	Global analysis.
psc26df1.inp	Submodel analysis.

Abaqus/Explicit input files

submodelht_g_cpe6mt_xpl.inp	CPE6MT, global analysis.
submodelht_g_cps6mt_xpl.inp	CPS6MT, global analysis.
submodelht_s_cpe6mt_xpl.inp	CPE6MT, submodel analysis.
submodelht_s_cps6mt_xpl.inp	CPE6MT, submodel analysis.

DC3D8, C3D8RT elements

Problem description

Model: The geometry is taken from the patch test in *ec38dfp4.inp*.

The global model dimensions are 1 × 1 × 1. The submodel lies completely inside the global model.

Material:

Conductivity	4.85×10^{-4}
Density	0.283
Specific heat	0.116

Loading: Nodal temperatures of 0 and 1000 are prescribed on the planes y=0 and y=1, respectively.

Results and discussion

In Abaqus/Standard steady-state heat transfer analysis is performed. In Abaqus/Explicit a transient analysis is performed over a period of time in which the steady-state regime is reached. The nodal temperatures for the driven nodes in the submodel are correctly interpolated from the global model results.

Input files

Abaqus/Standard input files

pgc38dfs.inp	Global analysis.
psc38df1.inp	Submodel analysis.

Abaqus/Explicit input files

submodelht_g_c3d8rt_xpl.inp	C3D8RT, global analysis.
submodelht_s_c3d8rt_xpl.inp	C3D8RT, submodel analysis.

DCAX8, CAX4RT elements

Problem description

Model: The geometry is taken from the patch test in *ec28dfp4.inp*.

The planar dimensions for the global model are 0.24×0.12. One side of the submodel lies along the right-hand side of the global model, while the remaining three sides of the submodel lie completely inside the global model.

Material:

Conductivity	4.85×10^{-4}
Density	0.283
Specific heat	0.116

Loading: A body force flux is applied on the entire model. A radiation load with a reference sink temperature of 1000 and a radiation constant of $10. \times 10^{-13}$ is applied along the right edge. A nodal temperature of 900 is prescribed at nodes on the left edge and the middle nodes on the top and bottom edges.

Results and discussion

In Abaqus/Standard steady-state heat transfer analysis is performed. In Abaqus/Explicit a transient analysis is performed over a period of time in which the steady-state regime is reached. The nodal temperatures for the driven nodes in the submodel are correctly interpolated from the global model results.

Input files

Abaqus/Standard input files

pgca8dfs.inp	Global analysis.
psca8df1.inp	Submodel analysis.

Abaqus/Explicit input files

submodelht_g_cax4rt_xpl.inp	CAX4RT, global analysis.
submodelht_s_cax4rt_xpl.inp	CAX4RT, submodel analysis.

Coupled temperature-displacement submodeling

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

SC8RT
C3D8HT C3D8RHT C3D8RT C3D4T C3D6T C3D8T C3D20HT C3D20RHT C3D20RT C3D20T
CAX4HT CAX4RHT CAX3T CAX4T CAX4T CAX6MHT CAX6MT CAX8HT CAX8RHT CAX4RT CAX8RT CAX8T
CGAX3HT CGAX3T CGAX4HT CGAX4RHT CGAX4RT CGAX4T CGAX6MHT CGAX6MT CGAX8HT CGAX8RHT CGAX8RT CGAX8T
CPE4HT CPE4RHT CPE4T CPE6MHT CPE6MT CPE8HT CPE8RHT CPE8RT CPE3T CPE4RT CPE8T
CPEG3T CPEG4RHT CPEG4RT CPEG4T CPEG6MHT CPEG6MT CPEG8T
CPS4RT CPS3T CPS4T CPS6MT CPS8RT CPS8T

Features tested

The submodeling capability is applied to two-dimensional, three-dimensional, and axisymmetric continuum coupled temperature-displacement elements. General steps invoking the steady-state coupled temperature-displacement and the dynamic temperature-displacement procedures are used in Abaqus/Standard and Abaqus/Explicit, respectively, for both the global and submodel analyses.

Problem description

Model: All global models have dimensions 7.0 × 7.0 in the x–y or r–z plane. Each submodel has dimensions 5.0 × 5.0 in the x–y or r–z plane and occupies the lower right-hand corner of the corresponding global model. In all but the axisymmetric models, the out-of-plane dimension is 1.0. In axisymmetric models the structure analyzed is a hollow cylinder with an outer radius of 8.0.

Material: In Abaqus/Standard:

Young's modulus	30×10^6
Poisson's ratio	0.3
Coeff. of thermal expansion	1×10^{-5}
Thermal conductivity	3.77×10^{-5}
Specific heat	0.39
Density	82.9

In Abaqus/Explicit:

Young's modulus	110×10^9
Poisson's ratio	0.3

Coeff. of thermal expansion	1×10^{-3}
Thermal conductivity	390
Specific heat	384
Density	8900

Loading: In all Abaqus/Standard models a distributed flux of magnitude 0.3 is applied to the right face; in Abaqus/Explicit the flux magnitude is 0.5×10^4.

Boundary and initial conditions: In the global model fixed boundary conditions u_1=0 and u_2=0 are prescribed on the left and bottom faces, respectively. In three-dimensional models the additional constraints u_3=0 are applied to the nodes on the front and back faces. The initial temperature is zero everywhere, and fixed temperature boundary conditions are applied on the left face. In the submodel u_2=0 is prescribed everywhere on the bottom face, while degrees of freedom 1, 2, and 11 for the nodes on the top and left faces are being driven by the global solution. The mass scaling technique is used in the Abaqus/Explicit models to speed-up the analysis.

Results and discussion

In the global analyses the temperature field predicted by Abaqus varies linearly in the *x*-direction in nonaxisymmetric models and logarithmically in the *r*-direction in axisymmetric models. The predicted displacement field is nonuniform in all models. The Abaqus/Standard results depicted for the temperature and *x*- or *r*-displacement contour plots are shown below. For comparison purposes the temperature and displacement solutions predicted by the submodels are also presented in the same contour plots, and excellent agreement between the global and submodel results is obtained. Hence, the amplitudes of all driven variables in the submodel analysis are identified correctly in the global analysis file output and applied at the driven nodes in the submodel analysis.

Global and submodel analyses results for 4-node plane stress elements in Abaqus/Standard are shown in *Figure 1* and *Figure 2*.

Global and submodel Abaqus/Standard analyses results for 8-node plane strain elements are shown in *Figure 3* and *Figure 4*.

Global and submodel Abaqus/Standard analyses results for 8-node axisymmetric elements are shown in *Figure 5* and *Figure 6*.

Global and submodel Abaqus/Standard analyses results for 20-node brick elements (front face) are shown in *Figure 7* and *Figure 8*.

In Abaqus/Explicit the driven temperatures and displacements in the submodel are correctly interpolated from the global analysis file output. Each of the two-dimensional, three-dimensional, or axisymmetric submodels can be driven from any global model that has the same dimensionality. The results between the global model and submodel agree extremely well.

Input files

Abaqus/Standard input files

The following input files test the steady-state fully coupled thermal-stress procedure:

pgc38ths.inp	C3D8HT elements; global analysis.
psc38ths.inp	C3D8HT elements; submodel analysis.
pgc38tys.inp	C3D8RHT elements; global analysis.

psc38tys.inp	C3D8RHT elements; submodel analysis.
pgc38trs.inp	C3D8RT elements; global analysis.
psc38trs.inp	C3D8RT elements; submodel analysis.
pgc38tfs.inp	C3D8T elements; global analysis.
psc38tfs.inp	C3D8T elements; submodel analysis.
pgc3kths.inp	C3D20HT elements; global analysis.
psc3kths.inp	C3D20HT elements; submodel analysis.
pgc3ktys.inp	C3D20RHT elements; global analysis.
psc3ktys.inp	C3D20RHT elements; submodel analysis.
pgc3ktrs.inp	C3D20RT elements; global analysis.
psc3ktrs.inp	C3D20RT elements; submodel analysis.
pgc3ktfs.inp	C3D20T elements; global analysis.
psc3ktfs.inp	C3D20T elements; submodel analysis.
pgca4ths.inp	CAX4HT elements; global analysis.
psca4ths.inp	CAX4HT elements; submodel analysis.
pgca4tys.inp	CAX4RHT elements; global analysis.
psca4tys.inp	CAX4RHT elements; submodel analysis.
pgca4trs.inp	CAX4RT elements; global analysis.
psca4trs.inp	CAX4RT elements; submodel analysis.
pgca4tfs.inp	CAX4T elements; global analysis.
psca4tfs.inp	CAX4T elements; submodel analysis.
pgca6ths.inp	CAX6MHT elements; global analysis.
psca6ths.inp	CAX6MHT elements; submodel analysis.
pgca6tfs.inp	CAX6MT elements; global analysis.
psca6tfs.inp	CAX6MT elements; submodel analysis.
pgca8ths.inp	CAX8HT elements; global analysis.
psca8ths.inp	CAX8HT elements; submodel analysis.
pgca8tys.inp	CAX8RHT elements; global analysis.
psca8tys.inp	CAX8RHT elements; submodel analysis.
pgca8trs.inp	CAX8RT elements; global analysis.
psca8trs.inp	CAX8RT elements; submodel analysis.
pgca8tfs.inp	CAX8T elements; global analysis.
psca8tfs.inp	CAX8T elements; submodel analysis.
pgca3hhs.inp	CGAX3HT elements; global analysis.
psca3hhs.inp	CGAX3HT elements; submodel analysis.
pgca3hfs.inp	CGAX3T elements; global analysis.
psca3hfs.inp	CGAX3T elements; submodel analysis.
pgca4hhs.inp	CGAX4HT elements; global analysis.
psca4hhs.inp	CGAX4HT elements; submodel analysis.
pgca4hys.inp	CGAX4RHT elements; global analysis.
psca4hys.inp	CGAX4RHT elements; submodel analysis.
pgca4hrs.inp	CGAX4RT elements; global analysis.
psca4hrs.inp	CGAX4RT elements; submodel analysis.

pgca4hfs.inp	CGAX4T elements; global analysis.
psca4hfs.inp	CGAX4T elements; submodel analysis.
pgca6hhs.inp	CGAX6MHT elements; global analysis.
psca6hhs.inp	CGAX6MHT elements; submodel analysis.
pgca6hfs.inp	CGAX6MT elements; global analysis.
psca6hfs.inp	CGAX6MT elements; submodel analysis.
pgca8hhs.inp	CGAX8HT elements; global analysis.
psca8hhs.inp	CGAX8HT elements; submodel analysis.
pgca8hys.inp	CGAX8RHT elements; global analysis.
psca8hys.inp	CGAX8RHT elements; submodel analysis.
pgca8hrs.inp	CGAX8RT elements; global analysis.
psca8hrs.inp	CGAX8RT elements; submodel analysis.
pgca8hfs.inp	CGAX8T elements; global analysis.
psca8hfs.inp	CGAX8T elements; submodel analysis.
pgce4ths.inp	CPE4HT elements; global analysis.
psce4ths.inp	CPE4HT elements; submodel analysis.
pgce4tys.inp	CPE4RHT elements; global analysis.
psce4tys.inp	CPE4RHT elements; submodel analysis.
pgce4trs.inp	CPE4RT elements; global analysis.
psce4trs.inp	CPE4RT elements; submodel analysis.
pgce4tfs.inp	CPE4T elements; global analysis.
psce4tfs.inp	CPE4T elements; submodel analysis.
pgce4tfsg.inp	CPE4T elements; **SUBMODEL*, GLOBAL ELSET; global analysis.
psce4tfsg.inp	CPE4T elements; **SUBMODEL*, GLOBAL ELSET; submodel analysis.
pgce6ths.inp	CPE6MHT elements; global analysis.
psce6ths.inp	CPE6MHT elements; submodel analysis.
pgce6tfs.inp	CPE6MT elements; global analysis.
psce6tfs.inp	CPE6MT elements; submodel analysis.
pgce8ths.inp	CPE8HT elements; global analysis.
psce8ths.inp	CPE8HT elements; submodel analysis.
pgce8tys.inp	CPE8RHT elements; global analysis.
psce8tys.inp	CPE8RHT elements; submodel analysis.
pgce8trs.inp	CPE8RT elements; global analysis.
psce8trs.inp	CPE8RT elements; submodel analysis.
pgce8tfs.inp	CPE8T elements; global analysis.
psce8tfs.inp	CPE8T elements; submodel analysis.
pgcg3tfs.inp	CPEG3T elements; global analysis.
pscg3tfs.inp	CPEG3T elements; submodel analysis.
pgcg4tys.inp	CPEG4RHT elements; global analysis.
pscg4tys.inp	CPEG4RHT elements; submodel analysis.
pgcg4trs.inp	CPEG4RT elements; global analysis.
pscg4trs.inp	CPEG4RT elements; submodel analysis.
pgcg4tfs.inp	CPEG4T elements; global analysis.

pscg4tfs.inp	CPEG4T elements; submodel analysis.
pgcg4tfsg.inp	CPEG4T elements; **SUBMODEL*, GLOBAL ELSET; global analysis.
pscg4tfsg.inp	CPEG4T elements; **SUBMODEL*, GLOBAL ELSET; submodel analysis.
pgcg6ths.inp	CPEG6MHT elements; global analysis.
pscg6ths.inp	CPEG6MHT elements; submodel analysis.
pgcg6tfs.inp	CPEG6MT elements; global analysis.
pscg6tfs.inp	CPEG6MT elements; submodel analysis.
pgcg8tfs.inp	CPEG8T elements; global analysis.
pscg8tfs.inp	CPEG8T elements; submodel analysis.
pgcs4trs.inp	CPS4RT elements; global analysis.
pscs4trs.inp	CPS4RT elements; submodel analysis.
pgcs4tfs.inp	CPS4T elements; global analysis.
pscs4tfs.inp	CPS4T elements; submodel analysis.
pgcs6tfs.inp	CPS6MT elements; global analysis.
pscs6tfs.inp	CPS6MT elements; submodel analysis.
pgcs8trs.inp	CPS8RT elements; global analysis.
pscs8trs.inp	CPS8RT elements; submodel analysis.
pgcs8tfs.inp	CPS8T elements; global analysis.
pscs8tfs.inp	CPS8T elements; submodel analysis.

Abaqus/Explicit input files

submcoupledtmp_g_c3d4t_xpl.inp	C3D4T elements; global analysis.
submcoupledtmp_s_c3d4t_xpl.inp	C3D4T elements; submodel analysis.
submcoupledtmp_g_c3d6t_xpl.inp	C3D6T elements; global analysis.
submcoupledtmp_s_c3d6t_xpl.inp	C3D6T elements; submodel analysis.
submcoupledtmp_g_c3d8rt_xpl.inp	C3D8RT elements; global analysis.
submcoupledtmp_s_c3d8rt_xpl.inp	C3D8RT elements; submodel analysis.
submcoupledtmp_g_sc8rt_xpl.inp	SC8RT elements; global analysis.
submcoupledtmp_s_sc8rt_xpl.inp	SC8RT elements; submodel analysis.
submcoupledtmp_g_cax3t_xpl.inp	CAX3T elements; global analysis.
submcoupledtmp_s_cax3t_xpl.inp	CAX3T elements; submodel analysis.
submcoupledtmp_g_cax4rt_xpl.inp	CAX4RT elements; global analysis.
submcoupledtmp_s_cax4rt_xpl.inp	CAX4RT elements; submodel analysis.
submcoupledtmp_g_cax6mt_xpl.inp	CAX6MT elements; global analysis.
submcoupledtmp_s_cax6mt_xpl.inp	CAX6MT elements; submodel analysis.
submcoupledtmp_g_cpe3t_xpl.inp	CPE3T elements; global analysis.
submcoupledtmp_s_cpe3t_xpl.inp	CPE3T elements; submodel analysis.
submcoupledtmp_g_cpe4rt_xpl.inp	CPE4RT elements; global analysis.
submcoupledtmp_s_cpe4rt_xpl.inp	CPE4RT elements; submodel analysis.
submcoupledtmp_g_cpe6mt_xpl.inp	CPE6MT elements; global analysis.
submcoupledtmp_s_cpe6mt_xpl.inp	CPE6MT elements; submodel analysis.
submcoupledtmp_g_cps3t_xpl.inp	CPS3T elements; global analysis.

submcoupledtmp_s_cps3t_xpl.inp	CPS3T elements; submodel analysis.
submcoupledtmp_g_cps4rt_xpl.inp	CPS4RT elements; global analysis.
submcoupledtmp_s_cps4rt_xpl.inp	CPS4RT elements; submodel analysis.
submcoupledtmp_g_cps6mt_xpl.inp	CPS6MT elements; global analysis.
submcoupledtmp_s_cps6mt_xpl.inp	CPS6MT elements; submodel analysis.

Figures

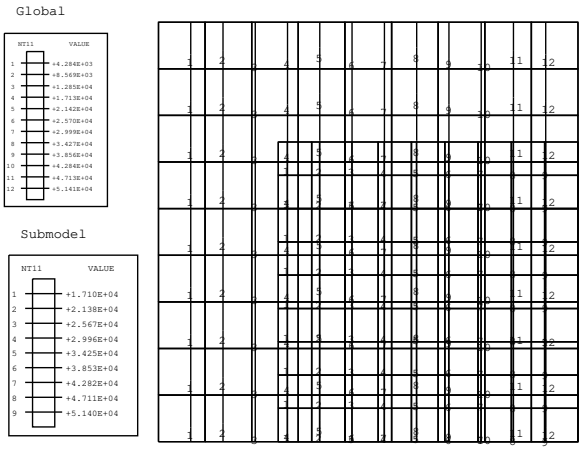


Figure 1: Temperature contours in global and submodels: 4-node plane stress.

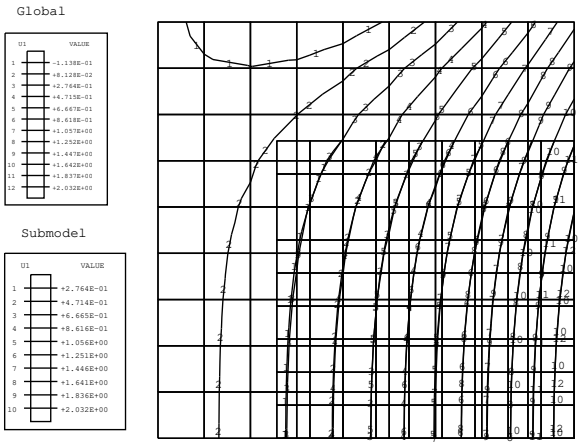


Figure 2: u_x contours in global and submodels: 4-node plane stress.

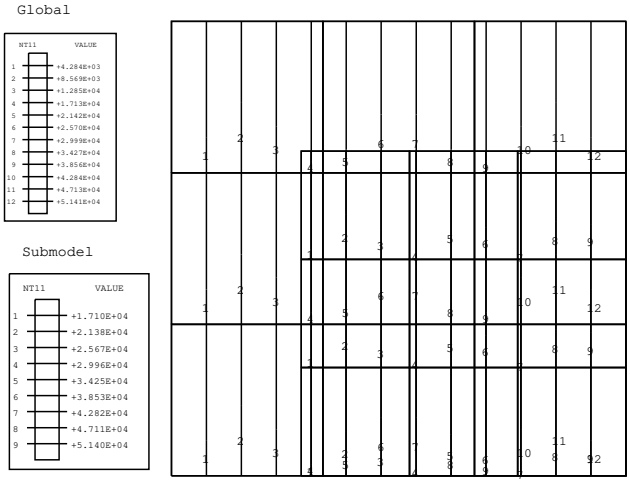


Figure 3: Temperature contours in global and submodels: 8-node plane strain.

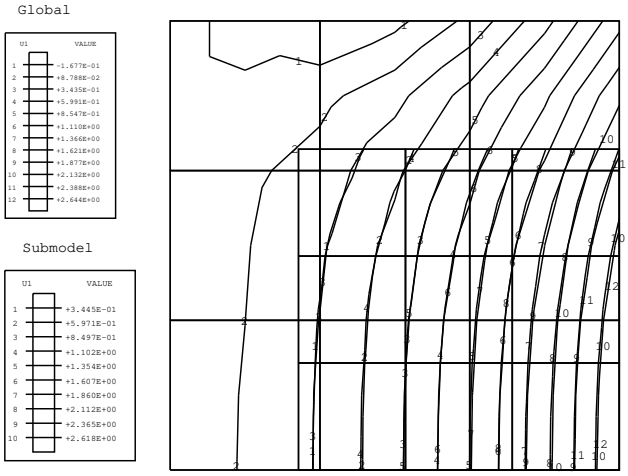


Figure 4: u_x contours in global and submodels: 8-node plane strain.

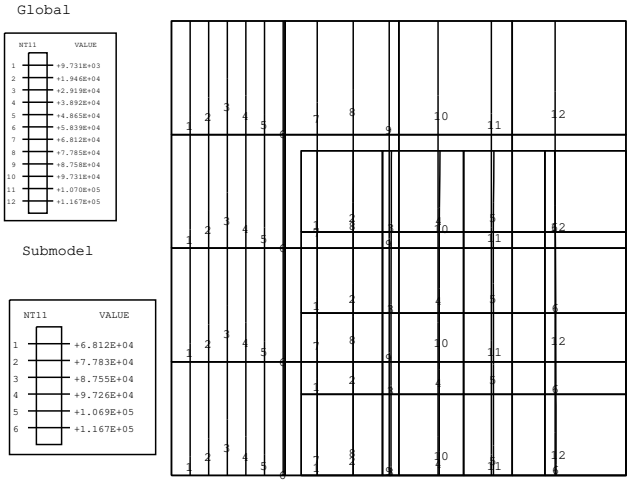


Figure 5: Temperature contours in global and submodels: 8-node axisymmetric.

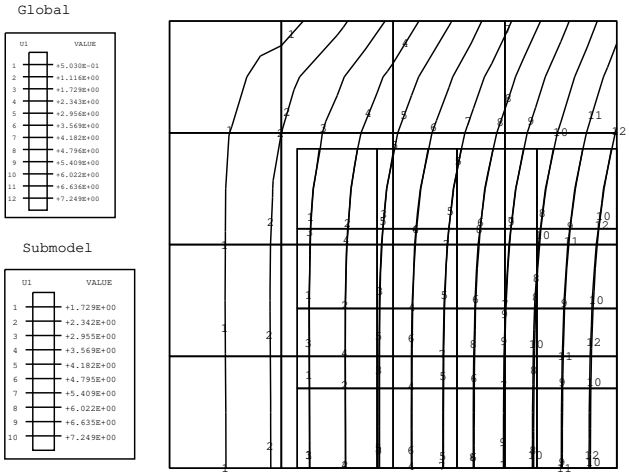


Figure 6: u_r contours in global and submodels: 8-node axisymmetric.

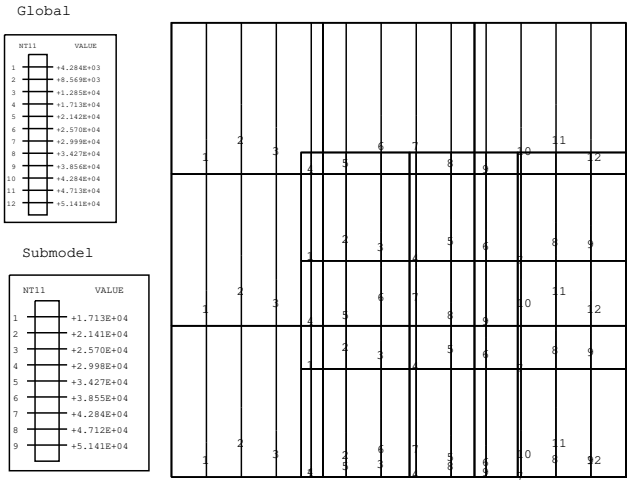


Figure 7: Temperature contours in global and submodels: 20-node brick.

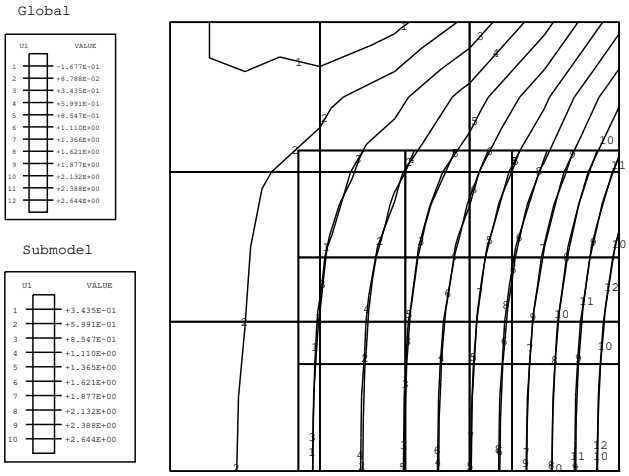


Figure 8: u_x contours in global and submodels: 20-node brick.

Pore pressure submodeling

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D8PH C3D8PT C3D10MP C3D20P C3D20RP
CAX4RP CAX6MP CAX8P CAX8RP
CPE4P CPE6MP CPE8P CPE8RP

Features tested

The submodeling capability is applied to two-dimensional, three-dimensional and axisymmetric continuum elements with pore pressure. General steps invoking the soils consolidation procedure are used for both the global and submodel analyses.

Problem description

Model: All global models have dimensions 3.0 × 5.0 in the *x*–*y* or *r*–*z* plane. Each submodel has dimensions 2.05 × 3.45 in the *x*–*y* or *r*–*z* plane and occupies the lower right-hand corner of the corresponding global model. In all but the axisymmetric models, the out-of-plane dimension is 1.0. In axisymmetric models the structure analyzed is a hollow cylinder with outer radius 5.0.

Material:

Young's modulus	100×10^6
Poisson's ratio	0.0
Permeability	1×10^{-5}
Density	1.4142

Loading: In all models, a distributed flow of magnitude 0.002 is applied to the right face, where the sink pore pressure is 14.7.

Boundary and initial conditions: In the global model, fixed boundary conditions $u_1 = 0$ and $u_2 = 0$ are prescribed on the left and bottom faces, respectively. In three-dimensional models the additional constraints $u_3 = 0$ are applied to the nodes on the front and back faces. The initial void ratio is unity everywhere and fixed pore pressure boundary conditions are applied on the left face. In the submodel, $u_2 = 0$ is prescribed everywhere on the bottom face, while degrees of freedom 1, 2 and 8 for the nodes on the top and left faces are being driven by the global solution.

Results and discussion

In the global analyses, the pore pressure field predicted by Abaqus varies linearly in the *x*-direction in nonaxisymmetric models and logarithmically in the *r*-direction in axisymmetric models. The predicted

displacement field is nonuniform in all models. These results are depicted in the pore pressure and *x*- or *r*-displacement contour plots shown below. For comparison purposes, the pore pressure and displacement solutions predicted by the submodels are also presented in the same contour plots and excellent agreement between global and submodel results is obtained. Hence, the amplitudes of all driven variables in the submodeled analysis are correctly identified in the global analysis file output and applied at the driven nodes in the submodel analysis.

Global and submodel analyses results for 8-node plane strain elements are shown in *Figure 1* and *Figure 2*.

Global and submodel analyses results for 8-node axisymmetric elements are shown in *Figure 3* and *Figure 4*.

Global and submodel analyses results for 20-node brick elements (front face) are shown in *Figure 5* and *Figure 6*.

Input files

The following input files test the transient soils consolidation procedure. Each test performs a single increment transient consolidation calculation for a time period of one.

pgc38phd.inp	C3D8PH elements; global analysis.
psc38phd.inp	C3D8PH elements; submodel analysis.
pgc3apkd.inp	C3D10MP elements; global analysis.
psc3apkd.inp	C3D10MP elements; submodel analysis.
pgc3kpfd.inp	C3D20P elements; global analysis.
psc3kpfd.inp	C3D20P elements; submodel analysis.
pgc3kprd.inp	C3D20RP elements; global analysis.
psc3kprd.inp	C3D20RP elements; submodel analysis.
pgca4prd.inp	CAX4RP elements; global analysis.
psca4prd.inp	CAX4RP elements; submodel analysis.
pgca6pkd.inp	CAX6MP elements; global analysis.
psca6pkd.inp	CAX6MP elements; submodel analysis.
pgca8pfd.inp	CAX8P elements; global analysis.
psca8pfd.inp	CAX8P elements; submodel analysis.
pgca8prd.inp	CAX8RP elements; global analysis.
psca8prd.inp	CAX8RP elements; submodel analysis.
pgce4pfd.inp	CPE4P elements; global analysis.
psce4pfd.inp	CPE4P elements; submodel analysis.
pgce6pkd.inp	CPE6MP elements; global analysis.
psce6pkd.inp	CPE6MP elements; submodel analysis.
pgce8pfd.inp	CPE8P elements; global analysis.
psce8pfd.inp	CPE8P elements; submodel analysis.
pgce8prd.inp	CPE8RP elements; global analysis.
psce8prd.inp	CPE8RP elements; submodel analysis.
ctp_gbmodel.inp	C3D8PT elements; global analysis.
ctp_sbmodel.inp	C3D8PT elements; submodel analysis.

Figures

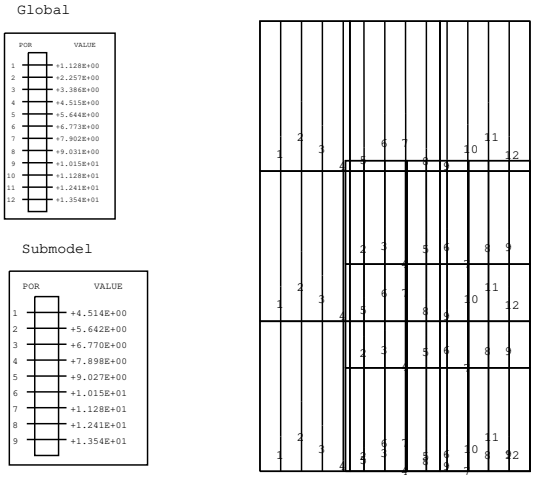


Figure 1: Pore pressure contours in global and submodels: 8-node plane strain.

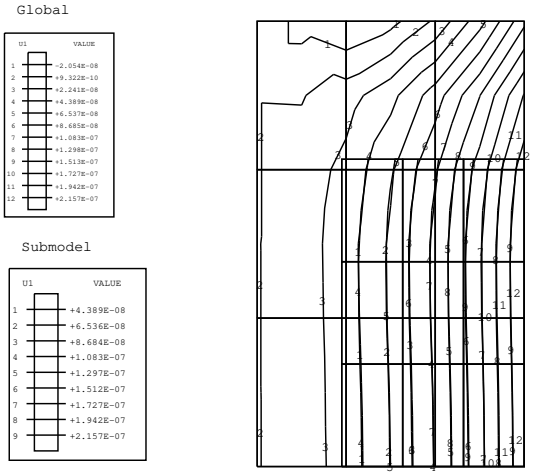


Figure 2: u_x contours in global and submodels: 8-node plane strain.

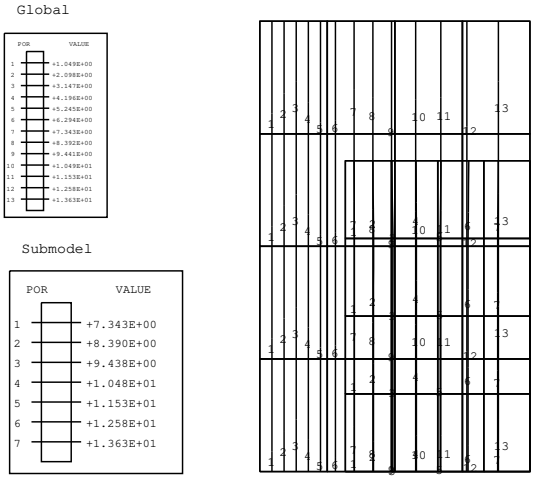


Figure 3: Pore pressure contours in global and submodels: 8-node axisymmetric.

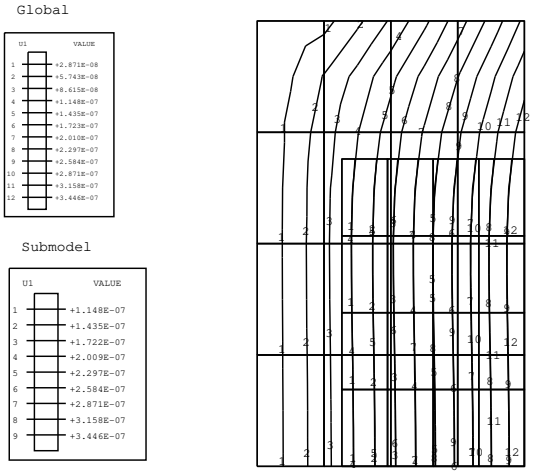


Figure 4: u_r contours in global and submodels: 8-node axisymmetric.

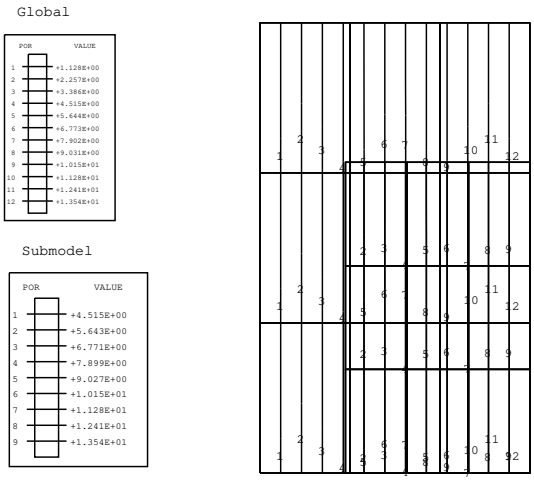


Figure 5: Pore pressure contours in global and submodels: 20-node brick.

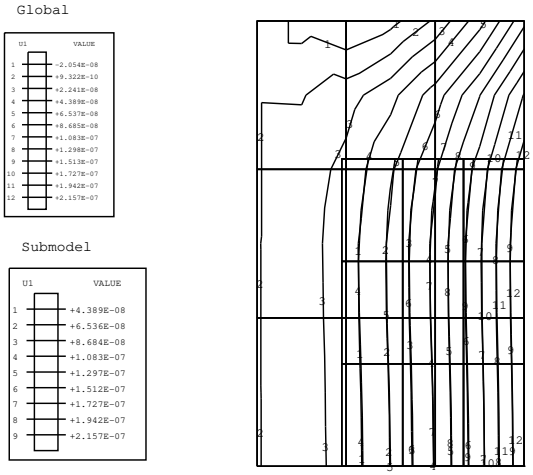


Figure 6: u_x contours in global and submodels: 20-node brick.

Piezoelectric submodeling

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

C3D8E elements

Elements tested

C3D8E CPE8E CPS8E

Features tested

The submodeling capability is applied to plane stress, plane strain, and three-dimensional solid piezoelectric elements. The global models consist of one element; the submodels model half of the global model and consist of two elements. Static and steady-state procedures are used.

Problem description

Model: The global model dimensions are 7 × 7 × 7. The submodel models the right-hand half of the global model.

Material:

Young's modulus	3×10^6
Poisson's ratio	0.3
Density	10
Coeff. of thermal expansion	0.0001
Dielectric	anisotropic (see input file)

Boundary conditions: The electric charge and the displacements u_1 and u_3 are zero on the $x = 0$ plane. The electric charge and the u_2 displacement are zero on the $y = 0$ plane.

Loading: The initial temperature is −10 at all nodes. In the first step a distributed charge of 3000 is applied on the $x = 7$ plane and the $y = 7$ plane. In the second step a temperature of 40 is applied to all nodes.

Results and discussion

The nodal displacements and electrical potential for the driven nodes in the submodels are correctly interpolated from the global model results.

Input files

pgc38efm.inp	Global analysis.
psc38ef1.inp	Submodel analysis.

CPE8E elements

Problem description

Model: The global model dimensions are 7 × 7 with a thickness of 1.0. The submodel models the right-hand half of the global model.

Material:

Young's modulus	3×10^6
Poisson's ratio	0.3
Density	5×10^{-5}
Dielectric	5.872×10^{-9}

Boundary conditions: The electric charge and the u_1 displacement are zero along the left-hand edge. Along the bottom edge the electric charge and the u_2 displacement are zero.

Loading: A distributed charge of 100 is applied on the right-hand edge. A concentrated charge of −150 is applied on the upper right-hand corner.

Results and discussion

The nodal displacements and electrical potential for the driven nodes in the submodels are correctly interpolated from the global model results.

Input files

pgce8efs.inp	Global analysis.
psce8ef1.inp	Submodel analysis.

CPS8E elements

Problem description

Model: The global model dimensions are 7 × 7 with a thickness of 1.0. The submodel models the right half of the global model.

Material:

Young's modulus	3×10^6
Poisson's ratio	0.3
Density	5×10^{-5}
Coeff. of thermal expansion	0.0001
Dielectric	5.872×10^{-9}

Boundary conditions: The electric charge and the u_1 displacement are zero along the left edge. Along the bottom edge the electric charge and the u_2 displacement are zero.

Loading: In the static step distributed charges of 2000 and 3000 are applied on the right-hand edge and top edge respectively. A concentrated charge of −150 is applied at the upper right-hand corner. In the steady-state step, distributed pressures of 200 and 300 are applied on the right-hand edge and top edge, respectively.

Results and discussion

The nodal displacements and electrical potential for the driven nodes in the submodels are correctly interpolated from the global model results.

Input files

pgcs8erm.inp	Global analysis.
pscs8er1.inp	Submodel analysis.
pgcs8ermgm.inp	Global analysis; multiple **SUBMODEL* options.
pscs8er1gm.inp	Submodel analysis; multiple **SUBMODEL* options.

Acoustic submodeling

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

AC2D3 AC2D4 AC2D4R AC2D6 AC2D8
AC3D4 AC3D5 AC3D6 AC3D8 AC3D8R AC3D10 AC3D15 AC3D20
ACAX3 ACAX4 ACAX4R ACAX6 ACAX8

Features tested

The submodeling capability is applied to an acoustic model of a duct. The global model is represented by either 20 linear elements or 10 quadratic elements along the lengthwise direction of the duct. An absorbing boundary condition is applied at one end of the duct; loads are applied to the other end. The submodel models the part of the duct close to the absorbing end and has a finer mesh than the global model. The driven nodes of the submodel lie along the global model element boundaries. Two-dimensional, three-dimensional, and axisymmetric models are tested for the driven nodes' acoustic pressure; the direct solution steady-state dynamic and direct-integration implicit dynamic procedures are used in Abaqus/Standard, and the explicit dynamic procedure is used in Abaqus/Explicit. The transient simulations are performed for period of time long enough to allow the wave to propagate past the end of the duct. Each element type used in the global model can be tested against each other element type of similar dimensionality in the submodel.

Problem description

Model: The two-dimensional and axisymmetric global models have dimensions of 1.0×10.0, and the three-dimensional global models have dimensions of $1.0 \times 10.0 \times 1.0$. In the two- and three-dimensional cases the submodel covers the end of the duct from 8.5 to 10; the axisymmetric submodel is from 8.0 to 10.0.

Material:

Bulk modulus of acoustic medium	1.42176×10^5
Density	1.293

Boundary conditions: In the global linear models the bottom surface is subjected to acoustic pressures of 1.0 at the corner nodes; in the Abaqus/Standard global quadratic models consistent loads corresponding to a uniform acoustic pressure load are applied. In the submodel analyses the boundary conditions are driven by the results from the global models.

Loading: The top of the acoustic medium has an impedance boundary condition with the proportionality factors between pressure and displacement equal to 2.3323×10^{-3}.

Results and discussion

The amplitudes of acoustic pressures and their phases are correctly identified in the global analysis file output and applied at the driven nodes in the submodel analysis.

Input files

Abaqus/Standard input files

Global analyses:

pgca3afd.inp	ACAX3 elements.
pgca4afd.inp	ACAX4 elements.
pgca6afd.inp	ACAX6 elements.
pgca8afd.inp	ACAX8 elements.
pgc23afd.inp	AC2D3 elements.
pgc24afd.inp	AC2D4 elements.
pgc26afd.inp	AC2D6 elements.
pgc28afd.inp	AC2D8 elements.
pgc34afd.inp	AC3D4 elements.
pgc35afd.inp	AC3D5 elements.
pgc36afd.inp	AC3D6 elements.
pgc38afd.inp	AC3D8 elements.
pgc3aafd.inp	AC3D10 elements.
pgc3fafd.inp	AC3D15 elements.
pgc3kafd.inp	AC3D20 elements.
submodelacoust_gd_acax4_std.inp	ACAX4 elements; global **DYNAMIC* analysis.
submodelacoust_gd_ac2d4_std.inp	AC2D4 elements; global **DYNAMIC* analysis.
submodelacoust_gd_ac3d8_std.inp	AC3D8 elements; global **DYNAMIC* analysis.

Submodel analyses:

psca3af1.inp	ACAX3 elements.
psca4af1.inp	ACAX4 elements.
psca6af1.inp	ACAX6 elements.
psca8af1.inp	ACAX8 elements.
psc23af1.inp	AC2D3 elements.
psc24af1.inp	AC2D4 elements.
psc26af1.inp	AC2D6 elements.
psc28af1.inp	AC2D8 elements.
psc34af1.inp	AC3D4 elements.
psc35af1.inp	AC3D5 elements.
psc36af1.inp	AC3D6 elements.
psc38af1.inp	AC3D8 elements.
psc3aaf1.inp	AC3D10 elements.
psc3faf1.inp	AC3D15 elements.

psc3kaf1.inp	AC3D20 elements.
submodelacoust_sd_acax4_std.inp	ACAX4 elements; submodel **DYNAMIC* analysis.
submodelacoust_sd_ac2d4_std.inp	AC2D4 elements; submodel **DYNAMIC* analysis.
submodelacoust_sd_ac3d8_std.inp	AC3D8 elements; submodel **DYNAMIC* analysis.

Abaqus/Explicit input files

Global analyses:

submodelacoust_g_acax3_xpl.inp	ACAX3 elements.
submodelacoust_g_acax4r_xpl.inp	ACAX4R elements.
submodelacoust_g_ac2d3_xpl.inp	AC2D3 elements.
submodelacoust_g_ac2d4r_xpl.inp	AC2D4R elements.
submodelacoust_g_ac3d4_xpl.inp	AC3D4 elements.
submodelacoust_g_ac3d6_xpl.inp	AC3D6 elements.
submodelacoust_g_ac3d8r_xpl.inp	AC3D8R elements.

Submodel analyses:

submodelacoust_s_acax3_xpl.inp	ACAX3 elements.
submodelacoust_s_acax4r_xpl.inp	ACAX4R elements.
submodelacoust_s_ac2d3_xpl.inp	AC2D3 elements.
submodelacoust_s_ac2d4r_xpl.inp	AC2D4R elements.
submodelacoust_s_ac3d4_xpl.inp	AC3D4 elements.
submodelacoust_s_ac3d6_xpl.inp	AC3D6 elements.
submodelacoust_s_ac3d8r_xpl.inp	AC3D8R elements.

Shell-to-solid submodeling

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

In-plane loading

Elements tested

C3D8I C3D8R C3D20R S3R S3RS S4 S4R S4RS S4RSW S8R STRI3

Features tested

The submodeling capability is tested on patches of shell elements, with 6 degrees of freedom per node, subject to in-plane loading. In Abaqus/Standard general static, static perturbation, dynamic, and steady-state dynamic procedures are used in various combinations for both the global and submodel analyses. A general, nonlinear static procedure (using NLGEOM) is also included in a separate global and submodel analysis. In Abaqus/Explicit an explicit dynamic procedure (with NLGEOM=NO) is used for both the global and the submodel analyses. The dynamic and explicit dynamic procedures are also tested using NLGEOM=YES in both the global and submodel analyses.

Problem description

Model: All global models have dimensions 0.24×0.12 in the x–y plane and use five section points through the thickness of 0.0125.

Material:

Young's modulus	1×10^6
Poisson's ratio	0.25
Density	1.0

Loading and boundary conditions: $u_x = A\,(x + y/2)$, $u_y = A\,(y + x/2)$ at all exterior nodes, and $u_z = 0$ at all nodes. $A = 10^{-3}$ in the first step and changes from step to step. In the solid submodel this boundary condition is applied to all faces except the face parallel to the y–z plane at $x = 0.12$. The latter is driven by the global model.

Results and discussion

The amplitudes of all driven variables in the submodel analysis are correctly identified in the global analysis file output and applied at the driven nodes in the submodel analysis.

Input files

Abaqus/Standard input files

pgsf3srmp.inp	S3R elements; global analysis.

psc38simp1.inp S3R/C3D8I elements; submodel analysis.
psc3ksrmp1.inp S3R/C3D20R elements; submodel analysis.

pgse4srmp.inp S4 elements; global analysis.
psc38simp5.inp S4/C3D8I elements; submodel analysis.
psc3ksrmp5.inp S4/C3D20R elements; submodel analysis.

pgse4srmpgm.inp S4 elements; multiple **SUBMODEL* options; global analysis.
psc38simp5gm.inp S4/C3D8I elements; multiple **SUBMODEL* options; submodel analysis.
psc3ksrmp5gm.inp S4/C3D20R elements; multiple **SUBMODEL* options; submodel analysis.

pgsf4srmp.inp S4R elements; global analysis.
psc38simp2.inp S4R/C3D8I elements; submodel analysis.
psc3ksrmp2.inp S4R/C3D20R elements; submodel analysis.

pgsf8srmp.inp S8R elements; global analysis.
psc38simp3.inp S8R/C3D8I elements; submodel analysis.
psc3ksrmp3.inp S8R/C3D20R elements; submodel analysis.

pgs63srmp.inp STRI3 elements; global analysis.
psc38simp4.inp STRI3/C3D8I elements; submodel analysis.
psc3ksrmp4.inp STRI3/C3D20R elements; submodel analysis.

General nonlinear static analyses (using NLGEOM)

pgsf3srsp.inp S3R elements; global analysis.
psc38sisp1.inp S3R/C3D8I elements; submodel analysis.
psc3ksrsp1.inp S3R/C3D20R elements; submodel analysis.

pgse4srsp.inp S4 elements; global analysis.
psc38sisp5.inp S4/C3D8I elements; submodel analysis.
psc3ksrsp5.inp S4/C3D20R elements; submodel analysis.

pgse4srspgm.inp S4 elements; multiple **SUBMODEL* options; global analysis.
psc38sisp5gm.inp S4/C3D8I elements; multiple **SUBMODEL* options; submodel analysis.
psc3ksrsp5gm.inp S4/C3D20R elements; multiple **SUBMODEL* options; submodel analysis.

pgsf4srsp.inp S4R elements; global analysis.
psc38sisp2.inp S4R/C3D8I elements; submodel analysis.
psc3ksrsp2.inp S4R/C3D20R elements; submodel analysis.

pgsf8srsp.inp S8R elements; global analysis.
psc38sisp3.inp S8R/C3D8I elements; submodel analysis.
psc3ksrsp3.inp S8R/C3D20R elements; submodel analysis.

pgs63srsp.inp STRI3 elements; global analysis.
psc38sisp4.inp STRI3/C3D8I elements; submodel analysis.
psc3ksrsp4.inp STRI3/C3D20R elements; submodel analysis.

Dynamic analyses (NLGEOM=YES)

substs_g_s4r_p_nl_std.inp	S4R element; global analysis.
substs_s_shel_c3d8_p_nl_std.inp	C3D8 elements; submodel analysis.

Abaqus/Explicit input files

NLGEOM=NO

substs_g_s3r_p_xpl.inp	S3R element; global analysis.
substs_g_s3rs_p_xpl.inp	S3RS element; global analysis.
substs_g_s4r_p_xpl.inp	S4R element; global analysis.
substs_g_s4rs_p_xpl.inp	S4RS element; global analysis.
substs_g_s4rsw_p_xpl.inp	S4RSW element; global analysis.
substs_s_shel_c3d8r_p_xpl.inp	C3D8R elements; submodel analysis.

NLGEOM=YES

substs_g_s3r_p_nl_xpl.inp	S3R element; global analysis.
substs_g_s3rs_p_nl_xpl.inp	S3RS element; global analysis.
substs_g_s4r_p_nl_xpl.inp	S4R element; global analysis.
substs_g_s4rs_p_nl_xpl.inp	S4RS element; global analysis.
substs_g_s4rsw_p_nl_xpl.inp	S4RSW element; global analysis.
substs_s_shel_c3d8r_p_nl_xpl.inp	C3D8R elements; submodel analysis.

Bending of cantilevered plate

Elements tested

C3D8I C3D8R C3D20R S3R S3RS S4 S4R S4RS S4RSW S8R STRI3

Features tested

The submodeling capability is tested on a flat plate with uniform geometry made up of various shell elements, with 6 degrees of freedom per node, at the global level and three-dimensional continuum elements at the submodel level, subject to a bending load. In Abaqus/Standard general static, static perturbation, dynamic, and steady-state dynamic procedures are used in various combinations for both the global and submodel analyses. A general, nonlinear static procedure (using NLGEOM) is also included in a separate global and submodel analysis. In Abaqus/Explicit an explicit dynamic procedure (with NLGEOM= NO) is used for both the global and the submodel analyses. The dynamic and explicit dynamic procedures are also tested using NLGEOM=YES in both the global and submodel analyses.

Problem description

Model: All global models have dimensions 10.0×3.0 in the x–z plane and use five section points through the thickness of 0.1.

Material:

Young's modulus	1×10^6
Poisson's ratio	0.3
Density	1.0

Loading and boundary conditions: The global model is constrained such that all displacement and rotation degrees of freedom for nodes along the y-axis are suppressed. All elements in the global model are then subject to a uniform pressure load in the positive z-direction. The magnitude of the pressure varies from step to step. In the solid submodel the pressure is applied to a surface that corresponds to the midsurface of the shell elements.

Results and discussion

The amplitudes of all driven variables in the submodel analysis are correctly identified in the global analysis file output and applied at the driven nodes in the submodel analysis. Contour plots of displacements and Mises stress obtained for the submodels agree well with the contour plots of displacements and Mises stress obtained for the same region in the global models.

Input files

Abaqus/Standard input files

pgsf3srmf.inp	S3R elements; global analysis.
psc38simf1.inp	S3R/C3D8I elements; submodel analysis.
psc3ksrmf1.inp	S3R/C3D20R elements; submodel analysis.
pgse4srmf.inp	S4 elements; global analysis.
psc38simf5.inp	S4/C3D8I elements; submodel analysis.
psc3ksrmf5.inp	S4/C3D20R elements; submodel analysis.
pgsf4srmf.inp	S4R elements; global analysis.
psc38simf2.inp	S4R/C3D8I elements; submodel analysis.
psc3ksrmf2.inp	S4R/C3D20R elements; submodel analysis.
pgsf8srmf.inp	S8R elements; global analysis.
psc38simf3.inp	S8R/C3D8I elements; submodel analysis.
psc3ksrmf3.inp	S8R/C3D20R elements; submodel analysis.
pgs63srmf.inp	STRI3 elements; global analysis.
psc38simf4.inp	STRI3/C3D8I elements; submodel analysis.
psc3ksrmf4.inp	STRI3/C3D20R elements; submodel analysis.

General nonlinear static analyses (using NLGEOM)

pgsf3srsf.inp	S3R elements; global analysis.
psc38sisf1.inp	S3R/C3D8I elements; submodel analysis.
psc3ksrsf1.inp	S3R/C3D20R elements; submodel analysis.
pgse4srsf.inp	S4 elements; global analysis.
psc38sisf5.inp	S4/C3D8I elements; submodel analysis.
psc3ksrsf5.inp	S4/C3D20R elements; submodel analysis.
pgsf4srsf.inp	S4R elements; global analysis.
psc38sisf2.inp	S4R/C3D8I elements; submodel analysis.
psc3ksrsf2.inp	S4R/C3D20R elements; submodel analysis.

pgsf8srsf.inp	S8R elements; global analysis.
psc38sisf3.inp	S8R/C3D8I elements; submodel analysis.
psc3ksrsf3.inp	S8R/C3D20R elements; submodel analysis.
pgs63srsf.inp	STRI3 elements; global analysis.
psc38sisf4.inp	STRI3/C3D8I elements; submodel analysis.
psc3ksrsf4.inp	STRI3/C3D20R elements; submodel analysis.

Dynamic analyses (NLGEOM=YES)

substs_g_s4r_b_nl_std.inp	S4R element; global analysis.
substs_s_shel_c3d8_b_nl_std.inp	C3D8 elements; submodel analysis.

Abaqus/Explicit input files

NLGEOM=NO

substs_g_s3r_b_xpl.inp	S3R element; global analysis.
substs_g_s3rs_b_xpl.inp	S3RS element; global analysis.
substs_g_s4r_b_xpl.inp	S4R element; global analysis.
substs_g_s4rs_b_xpl.inp	S4RS element; global analysis.
substs_g_s4rsw_b_xpl.inp	S4RSW element; global analysis.
substs_s_shel_c3d8r_b_xpl.inp	C3D8R elements; submodel analysis.

NLGEOM=YES

substs_g_s3r_b_nl_xpl.inp	S3R element; global analysis.
substs_g_s3rs_b_nl_xpl.inp	S3RS element; global analysis.
substs_g_s4r_b_nl_xpl.inp	S4R element; global analysis.
substs_g_s4rs_b_nl_xpl.inp	S4RS element; global analysis.
substs_g_s4rsw_b_nl_xpl.inp	S4RSW element; global analysis.
substs_s_shel_c3d8r_b_nl_xpl.inp	C3D8R elements; submodel analysis.

Bending of cantilevered half-cylinder

Elements tested

C3D8I C3D8R C3D20R S3R S3RS S4 S4R S4RS S4RSW S8R STRI3

Features tested

The submodeling capability is tested on a half-cylinder consisting of various shell elements, with 6 degrees of freedom per node, at the global level and three-dimensional continuum elements at the submodel level, subject to a bending load. In Abaqus/Standard general static, static perturbation, dynamic, and steady-state dynamic procedures are used in various combinations for both the global and submodel analyses. A general, nonlinear static procedure (using NLGEOM) is also included in a separate global and submodel analysis. In Abaqus/Explicit an explicit dynamic procedure (with NLGEOM=NO) is used for both the global and the submodel analyses. The dynamic and explicit dynamic procedures are also tested using NLGEOM=YES in both the global and submodel analyses.

Problem description

Model: The global models have a radius of 10 and a length of 20 and use five section points through a thickness of 0.2.

Material:

Young's modulus	1×10^7
Poisson's ratio	0.3
Density	0.001

Loading and boundary conditions: In the global model one end is completely constrained and a uniform upward pressure is applied to all the elements. The magnitude of the pressure is varied from step to step. In the solid submodel the pressure is applied on the lower face.

Results and discussion

The amplitudes of all driven variables in the submodel analysis are correctly identified in the global analysis file output and applied at the driven nodes in the submodel analysis. Contour plots of displacements and Mises stress obtained for the submodels agree well with the contour plots of displacements and Mises stress obtained for the same region in the global models.

Input files

Abaqus/Standard input files

General static, static perturbation, and steady-state dynamic analyses

pgsf3srmc.inp	S3R elements; global analysis.
psc38simc1.inp	S3R/C3D8I elements; submodel analysis.
psc3ksrmc1.inp	S3R/C3D20R elements; submodel analysis.
pgsf3srmcg.inp	S3R elements; **SUBMODEL*, GLOBAL ELSET; global analysis.
psc3ksrmc1g.inp	S3R/C3D20R elements; **SUBMODEL*, GLOBAL ELSET; submodel analysis.
pgse4srmc.inp	S4 elements; global analysis.
psc38simc5.inp	S4/C3D8I elements; submodel analysis.
psc3ksrmc5.inp	S4/C3D20R elements; submodel analysis.
pgsf4srmc.inp	S4R elements; global analysis.
psc38simc2.inp	S4R/C3D8I elements; submodel analysis.
psc3ksrmc2.inp	S4R/C3D20R elements; submodel analysis.
pgsf8srmc.inp	S8R elements; global analysis.
psc38simc3.inp	S8R/C3D8I elements; submodel analysis.
psc3ksrmc3.inp	S8R/C3D20R elements; submodel analysis.
pgs63srmc.inp	STRI3 elements; global analysis.
psc38simc4.inp	STRI3/C3D8I elements; submodel analysis.
psc3ksrmc4.inp	STRI3/C3D20R elements; submodel analysis.

General nonlinear static analyses (using NLGEOM)

pgsf3srsc.inp	S3R elements; global analysis.
psc38sisc1.inp	S3R/C3D8I elements; submodel analysis.
psc3ksrsc1.inp	S3R/C3D20R elements; submodel analysis.
pgsf3srscg.inp	S3R elements; **SUBMODEL*, GLOBAL ELSET; global analysis.
psc3ksrsc1g.inp	S3R/C3D20R elements; **SUBMODEL*, GLOBAL ELSET; submodel analysis.
pgse4srsc.inp	S4 elements; global analysis.
psc38sisc5.inp	S4/C3D8I elements; submodel analysis.
psc3ksrsc5.inp	S4/C3D20R elements; submodel analysis.
pgsf4srsc.inp	S4R elements; global analysis.
psc38sisc2.inp	S4R/C3D8I elements; submodel analysis.
psc3ksrsc2.inp	S4R/C3D20R elements; submodel analysis.
pgsf8srsc.inp	S8R elements; global analysis.
psc38sisc3.inp	S8R/C3D8I elements; submodel analysis.
psc3ksrsc3.inp	S8R/C3D20R elements; submodel analysis.
pgs63srsc.inp	STRI3 elements; global analysis.
psc38sisc4.inp	STRI3/C3D8I elements; submodel analysis.
psc3ksrsc4.inp	STRI3/C3D20R elements; submodel analysis.

Dynamic analyses (NLGEOM=YES)

substs_g_s4r_c_nl_std.inp	S4R element; global analysis.
substs_s_shel_c3d8_c_nl_std.inp	C3D8 elements; submodel analysis.

Abaqus/Explicit input files

NLGEOM=NO

substs_g_s3r_c_xpl.inp	S3R element; global analysis.
substs_g_s3rs_c_xpl.inp	S3RS element; global analysis.
substs_g_s4r_c_xpl.inp	S4R element; global analysis.
substs_g_s4rs_c_xpl.inp	S4RS element; global analysis.
substs_g_s4rsw_c_xpl.inp	S4RSW element; global analysis.
substs_s_shel_c3d8r_c_xpl.inp	C3D8R elements; submodel analysis.

NLGEOM=YES

substs_g_s3r_c_nl_xpl.inp	S3R element; global analysis.
substs_g_s3rs_c_nl_xpl.inp	S3RS element; global analysis.
substs_g_s4r_c_nl_xpl.inp	S4R element; global analysis.
substs_g_s4rs_c_nl_xpl.inp	S4RS element; global analysis.
substs_g_s4rsw_c_nl_xpl.inp	S4RSW element; global analysis.
substs_s_shel_c3d8r_c_nl_xpl.inp	C3D8R elements; submodel analysis.

Gasket submodeling

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

GK3D6 GK3D6T GK3D8 GK3D8T GKAX4 GKPE4 GKPS4

Features tested

The submodeling capability is applied to gasket elements. The general static procedure is used for both the global and submodel analyses.

Problem description

Model: All global models consist of a rectangular gasket, with a length of 3 and a thickness of 0.125. The three-dimensional models have a width of 1. The submodel input files model the right half of the global model. All nodes on the submodel are driven.

Material:

Membrane elastic modulus	68.7×10^3
Transverse shear stiffness	1.06×10^4
Poisson's ratio	0.3
Thickness behavior	Pressure of 2.75×10^4 at a closure of 1

Loading and boundary conditions: The gasket is loaded through contact with two rigid surfaces. The lower rigid surface is held fixed. The upper rigid surface is moved down and rotated to impart a spatially varying stress response in the gasket. Additional boundary conditions are applied to the gasket to suppress rigid body motion.

Results and discussion

The submodel stress distribution is confirmed to agree with the global model.

Input files

submodelgasket_gk3d6_g.inp	GK3D6 elements; global analysis.
submodelgasket_gk3d8_s.inp	GK3D8 elements; submodel analysis.
submodelgasket_gk3d8_g.inp	GK3D8 elements; global analysis.
submodelgasket_gk3d6_s.inp	GK3D6 elements; submodel analysis.
submodel_gk3d6t_g.inp	GK3D6T elements; global analysis.
submodel_gk3d8t_s.inp	GK3D8T elements; submodel analysis.

submodel_gk3d8t_g.inp	GK3D8T elements; global analysis.
submodel_gk3d6t_s.inp	GK3D6T elements; submodel analysis.
submodelgasket_gkax4_g.inp	GKAX4 elements; global analysis.
submodelgasket_gkax4_s.inp	GKAX4 elements; submodel analysis.
submodelgasket_gkax4n_g.inp	GKAX4N elements; global analysis.
submodelgasket_gkax4_s_n.inp	GKAX4 elements; submodel analysis.
submodelgasket_gkax4_solid_g.inp	GKAX4 elements using solid element numbering; global analysis.
submodelgasket_gkax4_s_solid.inp	GKAX4 elements; submodel analysis.
submodelgasket_gkpe4_g.inp	GKPE4 elements; global analysis.
submodelgasket_gkpe4_s.inp	GKPE4 elements; submodel analysis.
submodelgasket_gkps4_g.inp	GKPS4 elements; global analysis.
submodelgasket_gkps4_s.inp	GKPS4 elements; submodel analysis.

Miscellaneous submodeling tests

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Using different procedures between the global model and the submodel

Elements tested

CAX4R CPS3 CPS4R C3D8R C3D8RT

Features tested

The submodeling capability is applied to different procedures between the global model and the submodel. The global procedure can be performed in Abaqus/Explicit and the submodel procedure in Abaqus/Standard or vice versa. When appropriate, a submodel boundary condition is used to adjust the time variable of the driven nodes to match the submodel analysis step time.

Problem description

The first set of problems is based on the models that are described in *Two-dimensional continuum stress/displacement submodeling*. In the examples used here, however, each analysis has a second compression step. The global analysis is performed in Abaqus/Explicit, and the submodel analysis is performed in Abaqus/Standard. The step times of the analyses are different. Since the Abaqus/Explicit job is quasi-static and the Abaqus/Standard job is static, the step time from the global model is scaled to match the step time period of the submodel analysis.

The second set of tests is based on the models that are described in *Coupled temperature-displacement submodeling*. The global model uses C3D8R elements, and the problem is a stress/displacement analysis. The submodel uses C3D8RT elements, and it is a coupled temperature-displacement analysis. The validity of this submodeling analysis is based on the fact that the temperature effects are relatively small at the submodel level.

The last set of problems tests the direct-integration implicit dynamic procedure with submodeling. The global analysis is performed in Abaqus/Standard, and the corresponding submodeling analysis is performed in Abaqus/Explicit, or vice versa.

Results and discussion

All of the driven variables are interpolated correctly from the global analysis. *Figure 1* shows the effect of the TIMESCALE parameter on the amplitude formed at the driven nodes. If the analyses have the same step time, the two curves will be identical.

In the second and third set of tests, the results agree well between the global model and the submodel.

Input files

submproc_g_quasi2static_xpl.inp	Global, TIMESCALE parameter; Abaqus/Explicit quasi-static analysis.
submproc_s_quasi2static_std.inp	Submodel, TIMESCALE parameter; Abaqus/Standard static analysis.

submproc_s_quasi2static_std_sb.inp Submodel, TYPE=SURFACE parameter; Abaqus/Standard static analysis.

submproc_s_quasi2st_2nd_std.inp Submodel, TIMESCALE parameter; second-order elements; Abaqus/Standard static analysis.

submproc_g_dyn2tempdisp_xpl.inp Global stress/displacement analysis; Abaqus/Explicit analysis.

submproc_s_dyn2tempdisp_xpl.inp Submodel coupled temperature-displacement driven by the stress/displacement model; Abaqus/Explicit analysis.

submproc_s_dyn2tempdisp_std.inp Submodel coupled temperature-displacement driven by the stress/displacement model; Abaqus/Standard analysis.

submodelaxielem_cax4r_gd_xpl.inp Global **DYNAMIC* analysis; Abaqus/Explicit analysis.

submodelaxielem_cax4r_sd_std.inp Submodel **DYNAMIC* analysis; Abaqus/Standard analysis.

submodel2delem_cps4r_gd_std.inp Global **DYNAMIC* analysis; Abaqus/Standard analysis.

submodel2delem_cps4r_sd_xpl.inp Submodel **DYNAMIC* analysis; Abaqus/Explicit analysis.

Figures

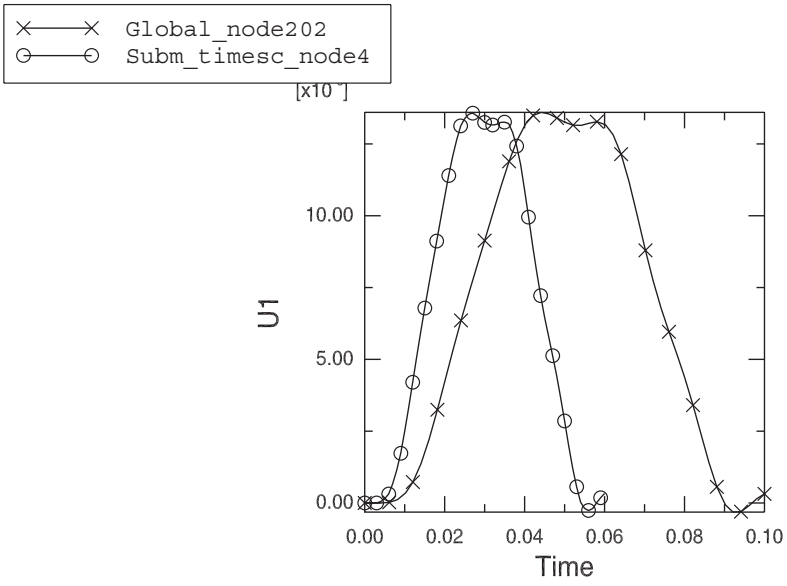


Figure 1: The effect of the TIMESCALE parameter on the displacement at a global node located very close to a submodel node.

Acoustic-to-structure submodeling

Elements tested

AC2D4R AC3D20 AC3D8 AC3D8R ACAX4R
CAX4R CPS4R C3D8R C3D8 C3D20
S4R S8R SAX1

Features tested

The submodeling capability is applied to the coupled acoustic-structural models. The global procedure is performed as a fully coupled acoustic-structural analysis in which the two media are coupled through the use of a tie constraint. Submodeling is performed on the structural component of the global model by using the acoustic pressure from a global acoustic structural model.

Problem description

In the global analysis acoustic pressure acts on either one or both sides of a flat panel. The flat panel is modeled using shell or solid elements. When the pressure acts on both sides of the panel, the correct side from which the acoustic pressures are to be interpolated is specified (see *Node-Based Submodeling*). The fluid and the structure in the global model have the material properties of water and steel, respectively. The submodel has the material properties of steel. For Abaqus/Standard the direct-integration implicit dynamic and the steady-state dynamic (direct and mode-based) procedures are used in separate tests.

Results and discussion

The loads resulting from the interpolated acoustic pressure from the global analysis are applied correctly on the structure for the single-sided and the double-sided pressure cases.

Input files

Abaqus/Standard Input files

ac2solid_g_c3d20_ac3d20_std.inp	Global analysis using **DYNAMIC*; fluid on one side; AC3D20 and C3D20 elements.
ac2solid_s_c3d20_ac3d20_std.inp	Submodel analysis using **DYNAMIC*; submodel driven on one side by acoustic pressure and on the second side by displacements; C3D20 elements.
ac2solid_g_c3d8_ac3d8_std.inp	Global analysis using **DYNAMIC*; fluid on one side; AC3D8 and C3D8 elements.
ac2solid_s_c3d8_ac3d8_std.inp	Submodel analysis using **DYNAMIC*; submodel driven on one side by acoustic pressure and on the second side by displacements; C3D8 elements.
ac2solid_g_s4_ac3d8_std.inp	Global analysis using **DYNAMIC*; fluid on two sides; S4 and AC3D8 elements.
ac2solid_s_s4_ac3d8_std.inp	Submodel analysis using **DYNAMIC*; submodel driven on both sides by the acoustic pressure; S4 elements.
ac2solid_s_s8r_ac3d8_std.inp	Submodel analysis using **DYNAMIC*; submodel driven on both sides by the acoustic pressure; S8R elements.
ac2solid_g_c3d8_ac3d8_ssd.inp	Global analysis using **STEADY STATE DYNAMICS*, DIRECT; fluid on one side; AC3D8 and C3D8 elements.
ac2solid_s_c3d8_ac3d8_ssd.inp	Submodel analysis using **STEADY STATE DYNAMICS*, DIRECT; submodel driven on one side by acoustic pressure and on the second side by displacements; C3D8 elements.

Abaqus/Explicit Input files

ac2solid_g_c3d8r_ac3d8r_xpl.inp	Global analysis; fluid on one side; AC3D8R and C3D8R elements.

ac2solid_s_c3d8r_ac3d8r_xpl.inp	Submodel analysis; submodel driven on one side by acoustic pressure and on the second side by displacements; C3D8R elements.
ac2solid_g_s4r_ac3d8r_xpl.inp	Global analysis; fluid on two sides; S4R and AC3D8R elements.
ac2solid_s_s4r_ac3d8r_xpl.inp	Submodel analysis; submodel driven on both sides by the acoustic pressure; S4R elements.
ac2solid_g_cax4r_acax4r_xpl.inp	Global analysis; fluid on one side; CAX4R and ACAX4R elements.
ac2solid_s_cax4r_acax4r_xpl.inp	Submodel analysis; submodel driven on one side by acoustic pressure and on the second side by displacements; CAX4R elements.
ac2solid_g_sax1_acax4r_xpl.inp	Global analysis; fluid on two sides; SAX1 and ACAX4R elements.
ac2solid_s_sax1_acax4r_xpl.inp	Submodel analysis; submodel driven on both sides by the acoustic pressure; SAX1 elements.
ac2solid_g_cps4r_ac2d4r_xpl.inp	Global analysis; fluid on one side; CPS4R and AC2D4R elements.
ac2solid_s_cps4r_ac2d4r_xpl.inp	Submodel analysis; submodel driven on one side by acoustic pressure and on the second side by displacements; CPS4R elements.

Intersection-only submodeling

Elements tested

C3D8 C3D8P C3D8R

Features tested

The submodeling capability is applied using the intersection-only feature, where nodes not found in the global model are ignored rather than labeled as errors.

Problem description

A simple model of a rectangular prism is used. The global model and submodel geometries are identical, but the submodel is shifted in space so that the intersection of the models represents a subset of the submodel geometry. All nodes in the submodel are identified as driven nodes.

Results and discussion

The results show that submodel boundary conditions are applied to driven nodes lying within the global model, while driven nodes lying outside the global model have no submodel boundary condition applied.

Input files

Abaqus/Standard Input files

subm_intonly_g_c3d8_std.inp	Global analysis using C3D8 elements.
subm_intonly_s_c3d8_std.inp	Submodel analysis using C3D8 elements and driven displacements.

subm_intonly_rs_c3d8_std.inp	Submodel restart analysis using C3D8 elements and driven displacements.
subm_intonly_g_c3d8p_std.inp	Global analysis using C3D8P elements.
subm_intonly_s_c3d8p_std.inp	Submodel analysis using C3D8P elements and driven displacements and pore pressures.

Abaqus/Explicit Input files

subm_intonly_g_c3d8r_xpl.inp	Global analysis using C3D8R elements.
subm_intonly_s_c3d8r_xpl.inp	Submodel analysis using C3D8R elements and driven displacements.

Node-based submodeling using field import

Elements tested

B21
C3D6 C3D8 C3D8PH C3D8R
S4 S4R

Features tested

These problems test submodeling for displacement, rotation, pressure, and temperature degrees of freedom.

Problem description

A subset of submodel verification problems in this section are changed to use the field import interface.

Results and discussion

Results are verified by comparison with existing results from submodel verification problems.

Input files

global_fieldimport.inp	Global model using C3D8 elements in a static procedure.
submodel_fieldimport.inp	Submodel using C3D8 elements in a static procedure.
fldimport_gbshell.inp	Global model using S4R elements in a static nonlinear procedure.
fldimport_sbshell.inp	Submodel using S4R elements in a static nonlinear procedure.
gbeam_fieldimport.inp	Global model using CPS4 elements in an implicit dynamic procedure.
sbeam_fieldimport.inp	Submodel using CPS4 elements in a static procedure.
global_c3d6_fieldimport.inp	Global model using C3D6 elements in an implicit dynamic procedure
submodel_c3d6_fieldimport.inp	Submodel using C3D6 elements in an implicit dynamic procedure.
xpl_global_fieldimport.inp	Global model using C3D8R elements in an explicit dynamic procedure.
xpl_submodel_fieldimport.inp	Submodel using C3D8R elements in an explicit dynamic procedure.
xpl_s4_global_fieldimport.inp	Global model using S4 elements in an explicit dynamic procedure.
xpl_s4_submodel_fieldimport.inp	Submodel using S4 elements in an explicit dynamic procedure.

Acoustic and shock analyses

In this section:

- *Volumetric drag*
- *Impedance boundary conditions*
- *Perfectly matched layers*
- *Transient acoustic wave propagation*
- *Adaptive meshing applied to coupled structural-acoustic problems*
- *CONWEP blast loading pressures*
- *Blast loading of a circular plate using the CONWEP model*

Volumetric drag

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

AC1D2 AC1D3
AC2D3 AC2D4 AC2D4R AC2D6 AC2D8
AC3D4 AC3D5 AC3D6 AC3D8 AC3D8R AC3D10 AC3D15 AC3D20
ACAX3 ACAX4 ACAX4R ACAX6 ACAX8

Features tested

Acoustic analysis in steady-state (direct and subspace-based) and transient analyses with high discontinuity in volumetric drag.

Problem description

The model consists of a tube of fluid 4 m long with a constant cross-sectional area. The tube lies horizontally (along the x-axis) and has a sound source at $x = 0$ m, which is given in the form of an inward volume acceleration. From $x = 0$ m to $x = 3$ m, the acoustic material in the tube is air with a bulk modulus of 1.424×10^5 N/m^2 and a density of 1.21 kg/m^3. The region from $x = 3$ m to $x = 4$ m is filled with a dissipative material with the same bulk modulus and density of air but with a volumetric drag of 10,000 Ns/m^4. The condition at $x = 4$ m is a closed end. The tube is modeled using 400 first-order or 200 second-order acoustic elements.

The speed of sound for these air constants is $c = 343$ m/s. At the highest frequency of 1100 Hz the wavelength is 0.312 m. The internodal interval (distance between nodes) for the meshes is always .01 m; therefore, at this frequency there are 30 first-order elements per wavelength or 15 second-order elements.

Both direct-solution and subspace-based steady-state dynamic analyses are performed in Abaqus/Standard over 3 frequencies ranging from 100 to 1100 Hz. The transient simulations are performed in Abaqus/Explicit using an excitation frequency of 100 Hz. Different excitation frequencies can be tested by changing the parameters defined in the input files. The transient analysis is also performed in Abaqus/Standard using the AC2D4 element for the purpose of providing a reference solution for Abaqus/Explicit.

Results and discussion

For Abaqus/Standard at the highest frequency the results with the second-order meshes lie within 0.1% of the analytical solution for the pressure and the phase in the air region. With the first-order meshes the results lie within 7%. As is to be expected, the second-order elements perform considerably better than first-order elements for the same number of degrees of freedom. Results for both types of mesh improve at lower frequencies (where there are more elements per wavelength).

The results from the transient analyses in Abaqus/Explicit agree very well with those obtained from Abaqus/Standard.

Input files

Abaqus/Standard input files

ec12afad.inp	AC1D2 elements.
ec13afad.inp	AC1D3 elements.
ec23afad.inp	AC2D3 elements.
ec24afad.inp	AC2D4 elements.
ec26afad.inp	AC2D6 elements.
ec28afad.inp	AC2D8 elements.
ec34afad.inp	AC3D4 elements.
ec35afad.inp	AC3D5 elements.
ec36afad.inp	AC3D6 elements.
ec38afad.inp	AC3D8 elements.
ec3aafad.inp	AC3D10 elements.
ec3fafad.inp	AC3D15 elements.
ec3kafad.inp	AC3D20 elements.
ec34afad_ams.inp	AC3D4 elements, Abaqus/AMS.
ec35afad_ams.inp	AC3D5 elements, Abaqus/AMS.
ec36afad_ams.inp	AC3D6 elements, Abaqus/AMS.
ec38afad_ams.inp	AC3D8 elements, Abaqus/AMS.
ec3aafad_ams.inp	AC3D10 elements, Abaqus/AMS.
ec3fafad_ams.inp	AC3D15 elements, Abaqus/AMS.
ec3kafad_ams.inp	AC3D20 elements, Abaqus/AMS.
eca3afad.inp	ACAX3 elements.
eca4afad.inp	ACAX4 elements.
eca6afad.inp	ACAX6 elements.
eca8afad.inp	ACAX8 elements.
ec12afaf.f	Fortran: analytical solution.
ec24afad_trans.inp	AC2D4 elements.

Abaqus/Explicit input files

eca3afad_trans_xpl.inp	ACAX3 elements.
eca4arad_trans_xpl.inp	ACAX4R elements.
ec23afad_trans_xpl.inp	AC2D3 elements.
ec24arad_trans_xpl.inp	AC2D4R elements.
ec34afad_trans_xpl.inp	AC3D4 elements.
ec36afad_trans_xpl.inp	AC3D6 elements.
ec38arad_trans_xpl.inp	AC3D8R elements.

Impedance boundary conditions

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Element-based and surface-based conditions

Elements tested

AC1D2 AC1D3
AC2D3 AC2D4 AC2D4R AC2D6 AC2D8
AC3D4 AC3D5 AC3D6 AC3D8 AC3D8R AC3D10 AC3D15 AC3D20
ACAX3 ACAX4 ACAX4R ACAX6 ACAX8

Features tested

Acoustic surface impedances on acoustic elements.

Problem description

The impedance boundary conditions are tested in this verification set. The model consists of a column of fluid 10 meters high with a cross-sectional area of 1 m. The first-order element models consist of 20 acoustic elements: 20 high and one in the cross-section. The second-order element models consist of 10 elements along the height direction.

One end of the column has a surface impedance imposed on it that is set equal to the characteristic impedance of the fluid column, $Z = \rho_f c$, where ρ_f is the density of the fluid and $c = \sqrt{K_f/\rho_f}$ is the speed of sound in the fluid. To simulate a nonreflecting boundary condition, $1/c_1 = 1/Z$ and $1/k_1 = 0$ are set with an impedance boundary condition. The material used in these tests is air with the following properties: density, $\rho_f = 1.293$ kg/m^3; bulk modulus, $K_f = 1.42176 \times 10^5$ N/m^2; and $1/c_1 = 1/Z = 2.3323 \times 10^{-3}$ m^{2}s/kg.

The other end of the column is excited by a harmonic pressure impulse of magnitude 1.0 N/m^2. A steady-state dynamic analysis is performed in Abaqus/Standard over a range of frequencies from 0 to 100 Hz. Transient simulations are also performed in Abaqus/Explicit using an excitation frequency of 100 Hz. Different excitation frequencies can be tested by changing the parameters defined in the input files. The solution should represent a steady-state unattenuated wave moving in the positive *y*-direction. No resonating frequencies should result; the maximum pressure throughout the column should consistently remain at a magnitude of 1.0 N/m^2, and the phase should drop by 2π radians over the distance of a wavelength, $\lambda = c/f$, where *f* is the excitation frequency in cycles per time.

Results and discussion

With the meshes used in these tests, the results lie within 8% of the analytical solution for the first-order models and within 2% of the analytical solution for the second-order models. Finer meshes yield more accurate results.

Input files

Abaqus/Standard input files

ec12afar.inp	AC1D2 elements.
ec13afar.inp	AC1D3 elements.
ec23afar.inp	AC2D3 elements.
ec24afar.inp	AC2D4 elements.
ec26afar.inp	AC2D6 elements.
ec28afar.inp	AC2D8 elements.
ec34afar.inp	AC3D4 elements.
ec35afar.inp	AC3D5 elements.
ec36afar.inp	AC3D6 elements.
ec38afar.inp	AC3D8 elements.
ec3aafar.inp	AC3D10 elements.
ec3fafar.inp	AC3D15 elements.
ec3kafar.inp	AC3D20 elements.
eca3afar.inp	ACAX3 elements.
eca4afar.inp	ACAX4 elements.
eca6afar.inp	ACAX6 elements.
eca8afar.inp	ACAX8 elements.

Abaqus/Explicit input files

eca3afar_trans_xpl.inp	ACAX3 elements.
eca4arar_trans_xpl.inp	ACAX4R elements.
ec23afar_trans_xpl.inp	AC2D3 elements.
ec24arar_trans_xpl.inp	AC2D4R elements.
ec34afar_trans_xpl.inp	AC3D4 elements.
ec36afar_trans_xpl.inp	AC3D6 elements.
ec38arar_trans_xpl.inp	AC3D8R elements.

Nonreflective boundaries

Elements tested

AC1D2 AC1D3
AC2D3 AC2D4 AC2D6 AC2D8
AC3D4 AC3D5 AC3D6 AC3D8 AC3D10 AC3D15 AC3D20

Features tested

Nonreflective boundaries on each of the acoustic elements, using nonreflective impedance default conditions for steady-state dynamic analyses in Abaqus/Standard. All elements are tested using the direct-solution steady-state dynamic prodecure; the AC2D4, AC2D8, and AC3D8 elements are also tested using the subspace-based steady-state dynamic procedure.

Problem description

These tests model a sound source at $x = 0$ m in a tube with significant volumetric drag (air properties with $\gamma =$ 1400 Ns/m^4) and a nonreflective end condition at $x = 0.5$ m at a frequency of 100 Hz. The complex density of the acoustic medium is specified in a second part in these analyses. In each model the inward acceleration of the sound source is specified as the complex value $a = \omega i$, giving an inward velocity of 1 m/s. (The inward acceleration on a face is distributed to the nodes of the face as concentrated loads representing inward volume accelerations in the same way as pressure on a face would be distributed to the nodes of the face as concentrated loads representing nodal forces.) Because of the large drag, for good results at this frequency the constants $1/c_1$ and $1/k_1$ must both be nonzero and must be based on the complex impedance of the medium.

Results and discussion

The results are within 1% of the analytical results, which are given as comments in the input files. The analytical result for the high-drag tests predicts exponential decay of pressure magnitude and linear dependence of pressure phase.

Input files

ec12afaw.inp	AC1D2 elements.
ec13afaw.inp	AC1D3 elements.
ec23afaw.inp	AC2D3 elements.
ec24afaw.inp	AC2D4 elements.
ec26afaw.inp	AC2D6 elements.
ec28afaw.inp	AC2D8 elements.
ec34afaw.inp	AC3D4 elements.
ec35afaw.inp	AC3D5 elements.
ec36afaw.inp	AC3D6 elements.
ec38afaw.inp	AC3D8 elements.
ec3aafaw.inp	AC3D10 elements.
ec3fafaw.inp	AC3D15 elements.
ec3kafaw.inp	AC3D20 elements.
ec34afaw_ams.inp	AC3D4 elements, Abaqus/AMS.
ec36afaw_ams.inp	AC3D6 elements, Abaqus/AMS.
ec38afaw_ams.inp	AC3D8 elements, Abaqus/AMS.
ec3aafaw_ams.inp	AC3D10 elements, Abaqus/AMS.
ec3fafaw_ams.inp	AC3D15 elements, Abaqus/AMS.
ec3kafaw_ams.inp	AC3D20 elements, Abaqus/AMS.
ec34afaw_sim.inp	AC3D4 elements.
ec3aafaw_sim.inp	AC3D10 elements.

Improved planar boundary condition

Elements tested

AC3D8 AC3D4 AC3D5 AC3D6
AC2D4 AC2D3

AC3D8R AC2D4R

Features tested

Nonreflective boundaries on each of the acoustic elements, using the nonreflective impedance condition for transient dynamic analyses in Abaqus/Standard and Abaqus/Explicit. All elements are tested using either the direct-integration implicit dynamic procedure in Abaqus/Standard or the explicit dynamic procedure in Abaqus/Explicit.

Problem description

These tests model one-dimensional propagation of sound in situations where the acoustic waves exit the acoustic domain through oblique boundaries. Various elementary geometric shapes are tested. In all models sinusoidal acoustic pressure boundary conditions are applied on one face of the acoustic domain, resulting in one-dimensional acoustic wave propagation in the model. The models are created so as to force the acoustic waves to exit from the model via surfaces that possess either continuously varying normals or normals that are not oriented in the same direction as the propagation of the waves. On the exit surface an impedance condition is used. The objective in all the models tested is to ensure that the problem remains one-dimensional and that there is no reflection of the acoustic waves back into the domain from the oblique boundary.

Results and discussion

By studying the contour plots of the acoustic pressure (POR), it can be seen that the acoustic waves retain their directionality (one-dimensional and normal to the loading surface) even in the regions adjacent to the oblique boundary. For example, *Figure 1* shows the contours of acoustic pressure in the case of a wedge-shaped model (brick45.inp) at the end of the analysis. As can be seen, the acoustic waves exit the boundary of the domain in exactly the same manner as they would if the boundary were normal to the outgoing plane waves.

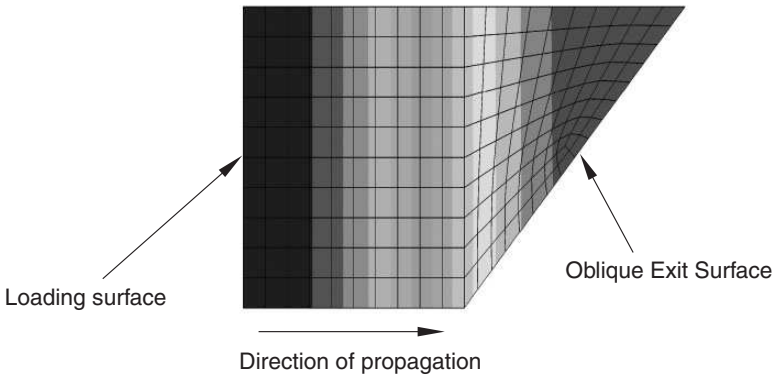


Figure 1: Acoustic pressure contours illustrating the effect of using an impedance condition to simulate a nonreflective boundary condition on an oblique surface.

Input files

Abaqus/Standard input files

brick45.inp	AC3D8 elements, oblique planar boundary.
bricksphere.inp	AC3D8 elements, spherical boundary.
quad45.inp	AC2D4 elements, oblique planar boundary.
quadcirc.inp	AC2D4 elements, circular boundary.
tet45.inp	AC3D4 elements, oblique planar boundary.

tetsphere.inp	AC3D4 elements, spherical boundary.
pyramid45.inp	AC3D5 elements, oblique planar boundary.
pyramidsphere.inp	AC3D5 elements, spherical boundary.
tri45.inp	AC2D3 elements, oblique planar boundary.
triacirc.inp	AC2D3 elements, circular boundary.
wed45.inp	AC3D6 elements, oblique planar boundary.

Abaqus/Explicit input files

brick45_xpl.inp	AC3D8 elements, oblique planar boundary.
bricksphere_xpl.inp	AC3D8 elements, spherical boundary.
quad45_xpl.inp	AC2D4 elements, oblique planar boundary.
quadcirc_xpl.inp	AC2D4 elements, circular boundary.
tet45_xpl.inp	AC3D4 elements, oblique planar boundary.
tetsphere_xpl.inp	AC3D4 elements, spherical boundary.
tri45_xpl.inp	AC2D3 elements, oblique planar boundary.
triacirc_xpl.inp	AC2D3 elements, circular boundary.
wed45_xpl.inp	AC3D6 elements, oblique planar boundary.

Acoustic interface elements

Elements tested

ASI1 ASI2 ASI3
ASI2A ASI3A
ASI4 ASI8
AC1D2 AC1D3
AC2D4 AC2D8
ACAX4 ACAX8
AC3D8 AC3D20

Features tested

Acoustic interface elements in Abaqus/Standard.

Problem description

For the ASI element tests the physical problem is similar to the nonreflective boundary test. Here, however, there is no volumetric drag, and a portion of the length of the body of air in the tube is modeled with truss elements. These are given Young's modulus and density to match the bulk modulus, $K_f = 1.424 \times 10^5$ N/m^2, and density, $\rho_f = 1.21$ kg/m^3, of air. The rest of the tube is modeled with acoustic elements that have the properties of air. Acoustic-structural coupling is set up between the structural region and the acoustic region using ASI elements, and a nonreflective end condition is applied.

This problem is analyzed for the one-dimensional case using ASI1 elements, for the two-dimensional case using ASI2 and ASI3 elements, for the axisymmetric case using ASI2A and ASI3A elements, and for the three-dimensional case using ASI4 and ASI8 elements. All the nodes in these models are constrained such that they have only the horizontal translation degree of freedom to simulate one-dimensional wave propagation.

Results and discussion

The results are within 1% of the analytical results, which are given as comments in the input files.

Input files

ec12afai.inp	ASI1/AC1D2 elements.
ec13afai.inp	ASI1/AC1D3 elements.
ec22afai.inp	ASI2/AC2D4 elements.
ec23afai.inp	ASI3/AC2D8 elements.
eca2afai.inp	ASI2A/ACAX4 elements.
eca3afai.inp	ASI3A/ACAX8 elements.
ec34afai.inp	ASI4/AC3D8 elements.
ec38afai.inp	ASI8/AC3D20 elements.

Impedance conditions on the semi-infinite sides of acoustic infinite elements

Elements tested

ACIN2D2 ACIN2D3
ACIN3D3 ACIN3D4 ACIN3D6 ACIN3D8
ACINAX2 ACINAX3

Features tested

Tabular impedance properties on each of the acoustic infinite elements for transient and steady-state dynamic analyses in Abaqus/Standard.

Problem description

These tests compare the behavior of acoustic infinite elements with and without impedance conditions defined on the semi-infinite sides. In all models the acoustic infinite elements are coupled directly to structural elements using steel material properties. The acoustic infinite elements use air properties and an impedance condition on one semi-infinite side with a tabular value corresponding to one-half the material impedance. In the steady-state dynamic analyses the frequency is varied from 1 to 200 Hz. In the transient dynamic analyses the elements are excited using a sinusoidal amplitude with an angular frequency of 5.

Results and discussion

The reductions in pressure amplitude due to the presence of the impedance condition on the acoustic infinite element sides are apparent in contour plots of the acoustic pressure.

Input files

Abaqus/Standard input files

ec2dafar_acin.inp	ACIN2D2 and ACIN2D3 elements, semi-infinite side impedance.
ec2dafas_acin.inp	ACIN2D2 and ACIN2D3 elements, semi-infinite side impedance.
ec3dafar_acin.inp	ACIN3D3, ACIN3D4, ACIN3D6, and ACIN3D8 elements, semi-infinite side impedance.
ec3dafas_acin.inp	ACIN3D3, ACIN3D4, ACIN3D6, and ACIN3D8 elements, semi-infinite side impedance.

ecaxafar_acin.inp	ACINAX2 and ACINAX3 elements, semi-infinite side impedance.
ecaxafas_acin.inp	ACINAX2 and ACINAX3 elements, semi-infinite side impedance.

Abaqus/Explicit input files

ec2dafas_acin_xpl.inp	ACIN2D2 elements, semi-infinite side impedance.
ec3dafas_acin_xpl.inp	ACIN3D3 and ACIN3D4 elements, semi-infinite side impedance.
ecaxafas_acin_xpl.inp	ACINAX2 elements, semi-infinite side impedance.

Perfectly matched layers

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

AC1D2 AC1D3 AC2D4 AC3D8 AC3D20

Features tested

This section provides basic verification tests for the perfectly matched layer elements that are used in Abaqus/Standard to truncate the acoustic infinite domain in a direct-solution steady-state dynamic analysis.

Problem description

The simple one-dimensional steady-state acoustic problem on the domain -1 to ∞ is solved using different acoustic elements. A source is applied at $x = -1$ of the form $u(-1) = e^{ik}$, where k is the wave number. The perfectly matched layer region is defined from 0 to 0.2 to truncate the domain. The perfectly matched layer coefficient that is used is 250. The wave number is $k = 12$, which gives the frequency of interest as 19.0986. The density of the acoustic medium is defined as 1, and the bulk modulus of the medium is 100. The source can be interpreted as an acoustic pressure boundary condition of 0.843853958 on the real part and –0.536572918 on the imaginary part. The end of the perfectly matched layer domain has a zero pressure boundary condition on both the real and imaginary parts.

Results and discussion

The analytical solution for this problem is given by ue^{-ikx}. The analytical solution and the solution computed by Abaqus/Standard are in good agreement.

Input files

acoustic_ssd_pml_1d2.inp	AC1D2 elements.
acoustic_ssd_pml_1d3.inp	AC1D3 elements.
acoustic_ssd_pml_2d4.inp	AC2D4 elements.
acoustic_ssd_pml_3d8.inp	AC3D8 elements.
acoustic_ssd_pml_3d20.inp	AC3D20 elements.

Transient acoustic wave propagation

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

AC1D2 AC1D3
AC2D3 AC2D4 AC2D4R AC2D6 AC2D8
AC3D4 AC3D5 AC3D6 AC3D8 AC3D8R AC3D10 AC3D15 AC3D20
ACAX3 ACAX4 ACAX4R ACAX6 ACAX8
ASI1 ASI2D2 ASI2D3
ASI3D3 ASI3D4 ASI3D6 ASI3D8
ASIAX2 ASIAX3

Features tested

Transient wave propagation in an acoustic medium with nonreflective end condition.

Problem description

The model consists of a column of fluid 1 m long with a constant cross-sectional area. The tube lies horizontally (along the *x*-axis), and the acoustic medium has a prescribed constant inward particle acceleration of $a_0 = 1$ m/s^2 at $x = 0$ m. A nonreflective boundary is specified at $x = 1$ m using nonreflective impedance conditions.

The acoustic material in the column is air with a bulk modulus $K_f = 1.424 \times 10^5$ N/m^2 and a density $\rho_f =$ 1.21 kg/m^3. The speed of sound is calculated as $c = \sqrt{\frac{K_f}{\rho_f}}$ = 343.05 m/s. The analytical result for the pressure is

$$p(x,t) = \frac{K_f a_0}{c^2}(ct - x), \qquad x < ct,$$
$$p(x,t) = 0, \qquad x \geq ct.$$

The column is modeled using either 100 first-order or 50 second-order acoustic elements. For each acoustic element tested, the acceleration is specified in each of two ways:

1. There is no ASI element or tie constraint, and an inward volume acceleration is specified on degree of freedom 8 as a concentrated load (“afav” files).
2. In Abaqus/Standard an ASI element is placed at $x = 0$ with its normal pointing into the fluid (this activates the displacement degree of freedom on the node at $x = 0$), and in Abaqus/Explicit a structural element with a tie constraint is used to define the interaction between the fluid and structure. An acceleration is prescribed directly with a boundary condition (“afas” files). In these cases the first time interval in the analysis is performed using boundary impedances; the analysis continues in time using surface impedances

A transient dynamic analysis is performed for a period long enough to allow the wave to propagate past the nonreflective boundary.

Results and discussion

Numerical results for the pressure distribution along the length of the acoustic medium are compared with the analytical results at time $t = 0.0044$ s.

In Abaqus/Standard for cases where the volume acceleration is specified ("afav" input files), the maximum errors are below 0.7%, except for linear tetrahedra, which are within 3% of the analytical results. For cases where a boundary acceleration is specified on the nodes of the ASI elements ("afas" input files), the maximum errors are below 4.2%, except for linear tetrahedra, which are within 5.9% of the analytical results.

In Abaqus/Explicit the results for pressure are within 0.2% of the analytical results for all tests, except for linear tetrahedra, which are within 3.2% of the analytical results.

Input files

Abaqus/Standard input files

ec12afav.inp	AC1D2 elements.
ec13afav.inp	AC1D3 elements.
ec23afav.inp	AC2D3 elements.
ec24afav.inp	AC2D4 elements.
ec26afav.inp	AC2D6 elements.
ec28afav.inp	AC2D8 elements.
ec34afav.inp	AC3D4 elements.
ec35afav.inp	AC3D5 elements.
ec36afav.inp	AC3D6 elements.
ec38afav.inp	AC3D8 elements.
ec3aafav.inp	AC3D10 elements.
ec3fafav.inp	AC3D15 elements.
ec3kafav.inp	AC3D20 elements.
eca3afav.inp	ACAX3 elements.
eca4afav.inp	ACAX4 elements.
eca6afav.inp	ACAX6 elements.
eca8afav.inp	ACAX8 elements.
ec12afas.inp	ASI1/AC1D2 elements.
ec13afas.inp	ASI1/AC1D3 elements.
ec23afas.inp	ASI2D2/AC2D3 elements.
ec24afas.inp	ASI2D3/AC2D4 elements.
ec26afas.inp	ASI2D3/AC2D6 elements.
ec28afas.inp	ASI2D3/AC2D8 elements.
ec34afas.inp	ASI3D4/AC3D4 elements.
ec34afas_po.inp	**POST OUTPUT* analysis.
ec36afas.inp	ASI3D3/ASI3D4/AC3D6 elements.
ec35afas.inp	ASI3D4/AC3D5 elements.
ec38afas.inp	ASI3D4/AC3D8 elements.
ec3aafas.inp	ASI3D6/AC3D10 elements.

ec3fafas.inp	ASI3D6/AC3D15 elements.
ec3kafas.inp	ASI3D8/AC3D20 elements.
eca3afas.inp	ASIAX2/ACAX3 elements.
eca4afas.inp	ASIAX2/ACAX4 elements.
eca6afas.inp	ASIAX3/ACAX6 elements.
eca8afas.inp	ASIAX3/ACAX8 elements.

Abaqus/Explicit input files

eca3afav_xpl.inp	ACAX3 elements.
eca4arav_xpl.inp	ACAX4R elements.
ec23afav_xpl.inp	AC2D3 elements.
ec24arav_xpl.inp	AC2D4R elements.
ec34afav_xpl.inp	AC3D4 elements.
ec36afav_xpl.inp	AC3D6 elements.
ec38arav_xpl.inp	AC3D8R elements.
eca3afas_xpl.inp	ACAX3 elements.
eca4aras_xpl.inp	ACAX4R elements.
ec23afas_xpl.inp	AC2D3 elements.
ec24aras_xpl.inp	AC2D4R elements.
ec34afas_xpl.inp	AC3D4 elements.
ec36afas_xpl.inp	AC3D6 elements.
ec38aras_xpl.inp	AC3D8R elements.

Adaptive meshing applied to coupled structural-acoustic problems

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Tire deflation with adaptive meshing

Elements tested

ACAX4 ACAX8
AC3D8 AC3D20

Features tested

Adaptive meshing, user-specified normal definition at a node, symmetric model generation, and symmetric boundary conditions.

Problem description

Model: A simple tire filled with air is analyzed, as shown in *Figure 1*. We model half of the cross-section. A negative pressure is applied to the inside of the structure, causing a significant decrease in the volume of the acoustic domain. We apply adaptive mesh smoothing after each converged structural load increment to compute a new acoustic mesh. We extract the eigenvalues of the coupled system after the preloading is applied. These eigenvalues are compared with the eigenvalues obtained in an independent analysis in which no adaptive mesh smoothing is performed. In this reference analysis both the acoustic mesh and structural mesh are defined in the displaced configuration. We apply an initial stress state that is in equilibrium with the pressure load so that no deformation takes place. The displaced configuration for the acoustic mesh is extracted from the results file. The displaced configuration for the structural mesh as well as the associated solution state that serves as the initial condition are obtained when the reference configuration is updated.

We also perform the same analysis using a three-dimensional model. We generate the model using symmetric model generation.

This example tests a number of adaptive mesh smoothing features. The adaptive mesh domain contains different node types, including interior nodes, corner nodes, surface nodes, nodes tied to the structure, as well as acoustic nodes that are connected using a tie constraint. The different updating rules associated with each of these node types are tested. In addition, application of the pressure load causes the volume of the acoustic elements to become negative. This, in turn, causes geometric feature changes (a corner develops) along the vertical surface. To avoid the development of corners, we transfer the structural displacement over a series of sub-increments to the acoustic domain. Adaptive meshing is applied after each sub-increment. The development of the corner can also be avoided by applying adaptive mesh controls. Both features are tested. Finally, the normal direction on the surface between the acoustic domain and structural domain is not computed correctly by Abaqus on a symmetry plane. The correct normal can be defined by using an alternative normal definition of contact surfaces or by applying symmetry boundary conditions. This example verifies that both these features are applied correctly during adaptive mesh smoothing.

Results and discussion

Figure 2 shows the displaced configuration. The eigenvalues agree closely with the reference solution, indicating that the geometry of the acoustic domain is updated correctly. The response of the system to harmonic excitation

is obtained using mode-based, direct-solution, and subspace-based steady-state dynamic analysis. The results agree well between the three analysis types.

Input files

am_tireair_acax4.inp	Axisymmetric tire-air model with ACAX4 elements and symmetric boundary conditions.
am_tireair_acax4_normal.inp	Axisymmetric tire-air model with ACAX4 elements and **NORMAL*.
am_tireair_acax4_tie.inp	Axisymmetric model with two acoustic regions connected using **TIE*.
am_tire_acax4.inp	Axisymmetric tire problem used as base state for the reference solution.
am_tireair_acax4_ver.inp	Axisymmetric tire-air problem used as a reference solution.
am_tireair_ac3d8.inp	Three-dimensional tire-air interaction with AC3D8 elements.
am_tire_ac3d8.inp	Three-dimensional tire problem used as base state for obtaining the reference solution.
am_tireair_ac3d8_ver.inp	Reference solution for three-dimensional model.
am_tireair_acax8.inp	Axisymmetric tire-air interaction with ACAX8 elements.
am_tire_acax8.inp	Axisymmetric tire problem used as base state for reference solution.
am_tireair_acax8_ver.inp	Axisymmetric model used as reference solution for second-order elements.
am_tireair_ac3d20.inp	Three-dimensional model with AC3D20 elements.

Figures

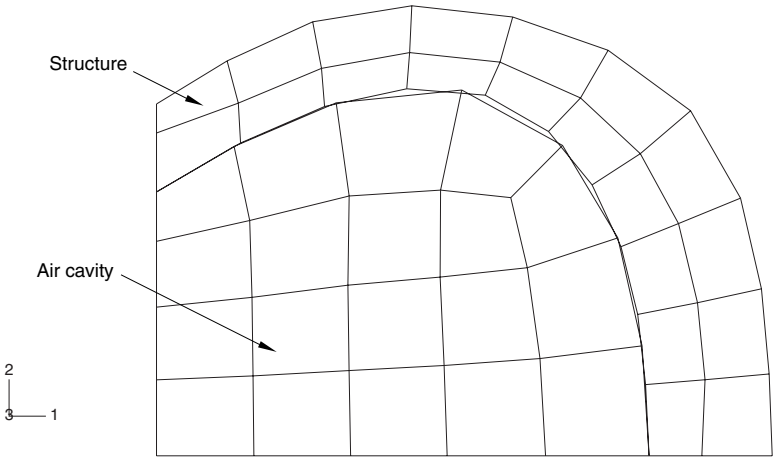


Figure 1: Initial tire-air mesh.

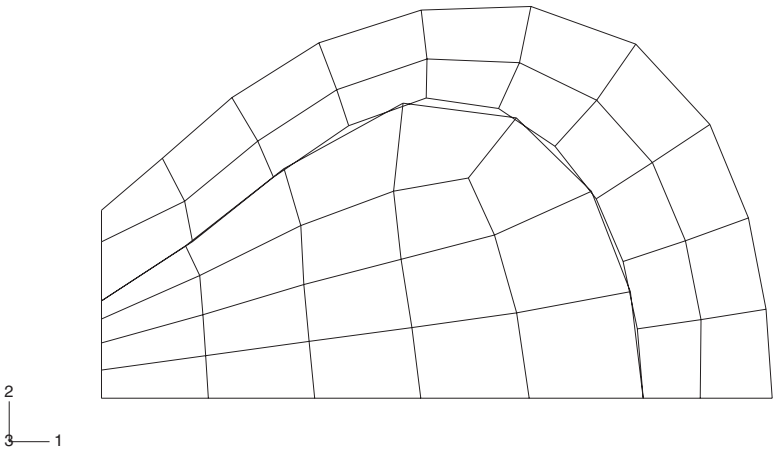


Figure 2: Deformed tire-air mesh.

Adaptive meshing applied to rigid body motion of a ring in a tank

Elements tested

ACAX3 ACAX6
ACAX4 ACAX8
AC3D8

Features tested

Adaptive mesh controls, adaptive mesh constraints, rigid body motion, and nodal transformations.

Problem description

This example consists of a circular structure filled with fluid. The structure is contained in a tank filled with fluid, as shown in *Figure 3*. A rigid body motion is applied to the structure, resulting in deformation of the fluid in the tank; while the fluid contained in the structure undergoes rigid body motion with the structure.

This example verifies a number of adaptive mesh smoothing features. To accommodate the large geometry changes of the fluid in the tank, nodes must slide along the vertical exterior surfaces of the tank. However, when the default adaptive mesh smoothing algorithm is applied to the exterior boundary region, no update takes place along the surface. This restricts the overall deformation of the acoustic domain. The reason for this is that the forcing function that drives adaptive smoothing is the displacement of the structure. Since the exterior of the acoustic surface is not connected to the structure, and since the update of a surface node is based entirely on the configuration of neighboring surface nodes, the exterior nodes decouple from the remaining nodes in the adaptive mesh smoothing equations. As a consequence, the exterior surface nodes are not updated. To overcome this problem, we define adaptive mesh constraints to specify a vertical displacement on two midsurface nodes as shown in *Figure 3*. We also use adaptive mesh controls to ensure that no geometric features develop on this sliding boundary.

This example further tests the different types of adaptive mesh smoothing rules applied to different element types, as well as the nodal transformations applied to different node types.

Results and discussion

Figure 4 shows the displaced configuration. The interior fluid domain undergoes rigid body motion without significant distortion.

Input files

am_tank_acax4.inp	ACAX4 elements with **TRANSFORM* applied on the interior nodes.
am_tank_acax8.inp	ACAX8 elements with **TRANSFORM* applied on the surface nodes.
am_tank_acax3.inp	ACAX3 elements.
am_tank_acax6.inp	ACAX6 elements.
am_tank_ac3d8.inp	AC3D8 elements.

Figures

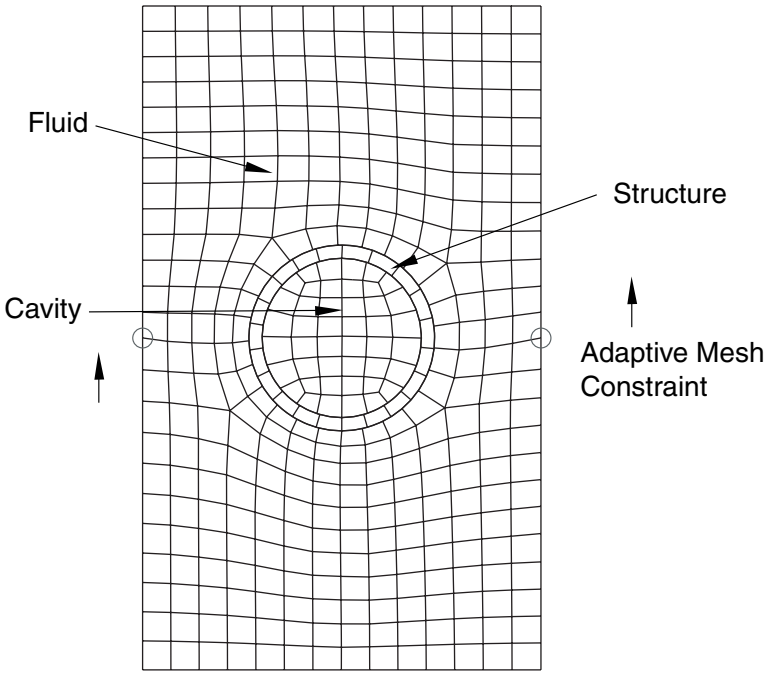


Figure 3: Initial tank mesh.

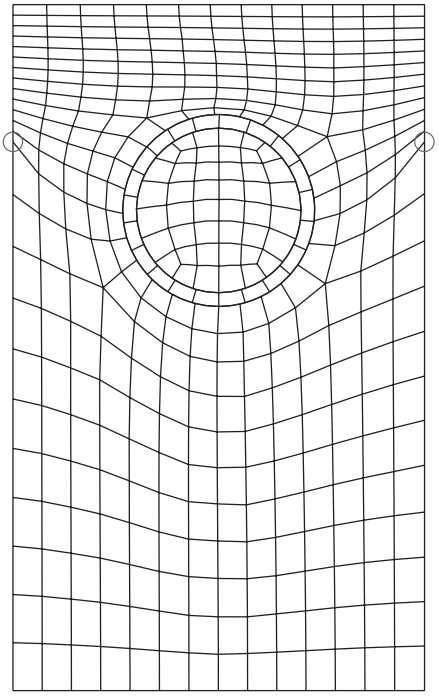


Figure 4: Tank mesh with rigid body displacement.

Rigid body motion of box filled with air

Elements tested

AC2D4 AC2D8
AC3D4

Features tested

Eigenfrequency extraction and rigid body motion.

Problem description

This example consists of a box filled with fluid, as shown in *Figure 5*. A large rigid body rotation is applied to the structure.

The example verifies that the geometric quantities associated with the fluid are updated correctly during adaptive mesh smoothing. We extract eigenvalues of the coupled system before and after the rigid body motion is applied. Since the rigid body motion is applied so that no strain develops in the structure, the eigenvalues before and after the loading must be identical.

Results and discussion

Figure 6 shows the displaced configuration. The acoustic mesh undergoes large rigid body motion without significant distortion of the mesh. The eigenvalues before the structural load is applied are identical to the eigenvalues obtained after application of the load, indicating that the geometric quantities in the acoustic domain are updated correctly.

Input files

am_box_ac2d4.inp	AC2D4 elements.

am_box_ac2d8.inp	AC2D8 elements.
am_box_ac3d4.inp	AC3D4 elements.

Figures

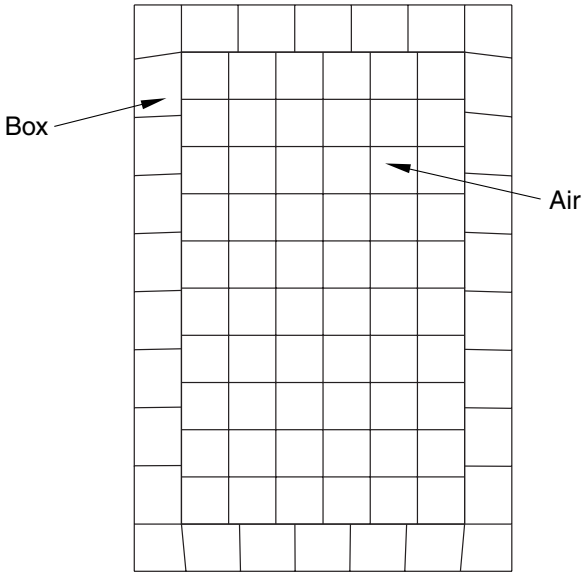


Figure 5: Initial configuration.

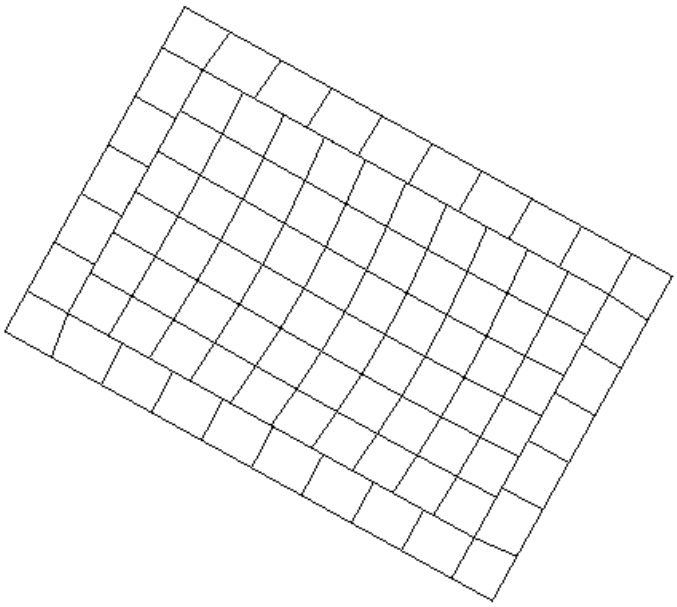


Figure 6: Displaced configuration.

CONWEP blast loading pressures

Products: Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D4 C3D6 C3D8 C3D10M
S3R S4RS

Features tested

Incident, reflected, and total pressures of the CONWEP model.

Problem description

Pressure output from the CONWEP blast loading model is verified in this set of tests. In each test all degrees of freedom of the loading surface are constrained, and the pressure is obtained by summing the reaction forces at all the nodes on the surface and dividing the sum by the surface area.

The pressure calculated from the reaction forces corresponds to the total pressure, which is a function of the incident pressure, the reflected pressure, and the incident angle. The total pressure corresponds to the incident pressure and the reflected pressure when the loading surface is given at incident angles of 180° and 0°, respectively. The variation of the incident and reflected pressures with range is verified in the range tests, in which loading surfaces at incident angles of 180° and 0° are placed at various distances from the source. The variation of the total pressure with the incident angle is verified in the angle test, in which multiple loading surfaces located at the same distance from the source are given at different incident angles.

Unit conversion is verified in the unit conversion tests, in which non-SI mass units (for the charge) and analysis units are used. Units of ton-mm-sec-MPa and lb-ft-sec-psf are considered in the tests.

For shell elements CONWEP blast loading pressure can be applied to both the SPOS and SNEG faces of the elements. Double-sided loading is verified in the test in which doubled-sided loading surfaces are orientated at incident angles of 0°, 90°, 180°, and 270°.

Results and discussion

The history of the incident and reflected pressures at various distances from the source can be computed from the following quantities: maximum incident and reflected pressures, arrival time, positive phase duration, and decay coefficients. For the purpose of verification these quantities are calculated independently for each of the range tests using empirical formulas of the CONWEP model given in Appendix A of *Randers-Pehrson and Bannister (1997)*. The results from the range tests match the results from independent calculations exactly in all cases.

For the angle test the results of the maximum total pressure at various incident angles satisfy the equations for the total pressure given as a function of the incident pressure, the reflected pressure, and the incident angle in *Acoustic and Shock Loads*.

For the unit conversion tests the results of the incident and reflected pressures, when converted to SI units, are identical to the results from a similar test that uses SI units.

For the double-sided loading test zero total pressure is obtained for the surfaces with incident angles 90° and 270°. This result is correct since, in both orientations, the pressure on the SPOS and SNEG faces is equal to the incident pressure but the pressure loads act in the opposite directions. The total pressure for the surfaces with incident angles of 0° and 180° is equal in magnitude but opposite in sign. This result is correct since, in both orientations, the pressure is equal to the difference between the reflected pressure and the incident pressure and the pressure loads act in the opposite directions.

Input files

Element tests

airblast_c3d4_pressures.inp	Air blast, C3D4 elements, range R, angles 0° and 180°.
airblast_c3d6_pressures.inp	Air blast, C3D6 elements, range R, angles 0° and 180°.
airblast_c3d8_pressures.inp	Air blast, C3D8 elements, range R, angles 0° and 180°.
airblast_c3d10m_pressures.inp	Air blast, C3D10M elements, range R, angles 0° and 180°.

Range tests

airblast_s3r_pressures.inp	Air blast, S3R elements, range R, angles 0° and 180°.
airblast_s4rs_pressures.inp	Air blast, S4RS elements, range R, angles 0° and 180°.
airblast_s4rs_2R.inp	Air blast, S4RS elements, range 2R, angles 0° and 180°.
airblast_s4rs_4R.inp	Air blast, S4RS elements, range 4R, angles 0° and 180°.
airblast_s4rs_8R.inp	Air blast, S4RS elements, range 8R, angles 0° and 180°.
airblast_s4rs_16R.inp	Air blast, S4RS elements, range 16R, angles 0° and 180°.
surfaceblast_s4rs_R.inp	Surface blast, S4RS elements, range R, angles 0° and 180°.
surfaceblast_s4rs_2R.inp	Surface blast, S4RS elements, range 2R, angles 0° and 180°.
surfaceblast_s4rs_4R.inp	Surface blast, S4RS elements, range 4R, angles 0° and 180°.
surfaceblast_s4rs_8R.inp	Surface blast, S4RS elements, range 8R, angles 0° and 180°.
surfaceblast_s4rs_16R.inp	Surface blast, S4RS elements, range 16R, angles 0° and 180°.

Angle test

airblast_s4rs_angle.inp	Air blast, S4RS elements, range R, angles 0° to 330° in increments of 30°.

Unit conversion tests

airblast_s4rs_ton_mm_mpa.inp	Air blast, S4RS elements, range R, angles 0° and 180°, units in tons, mm, sec, and MPa.
airblast_s4rs_lb_ft_psf.inp	Air blast, S4RS elements, range R, angles 0° and 180°, units in lb, ft, sec, and psf.

Double-sided loading test

airblast_s4rs_dble_side.inp	Air blast, S4RS elements, range R, angles 0°, 90°, 180°, and 270°, double-sided loading.

References

- Randers-Pehrson, G., and K. Bannister, “Airblast Loading Model for DYNA2D and DYNA3D,” Army Research Laboratory, ARL-TR-1310, March 1997.

Blast loading of a circular plate using the CONWEP model

Products: Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

S3R S4R

Features tested

Dynamic response to blast loading using the CONWEP model.

Problem description

A circular plate is subject to blast loading as a result of detonation of 50 kg of TNT 0.5 m directly above the center of the plate. The plate has a radius of 1 m and a thickness of 0.05 m. One-quarter of the plate is modeled using shell elements, with fully built-in boundary conditions applied along the circular edge and symmetry boundary conditions at the symmetry planes. Air blast CONWEP loading is applied on the top surface of the plate. The density of the plate material is 7850 kg/m^3, and the elastic material properties are Young's modulus of 210 GPa and Poisson's ratio of 0.28. The plastic behavior is modeled with a strain-rate insensitive isotropic hardening bilinear model, with yield stress of 1000 MPa and hardening modulus of 2 GPa. A dynamic analysis is performed for a period of .004 seconds.

Results and discussion

The history of the deflection at the center of the plate modeled using either S3R or S4R elements follows closely the result reported in *Neuberger et al. (2007)*. In addition, the history of the Mises stress at the SNEG location of the shell element at the plate center is consistent with the history of the effective stress given in the above reference.

Input files

airblast_s3r_circular_plate.inp	Circular plate modeled with S3R elements.
airblast_s4r_circular_plate.inp	Circular plate modeled with S4R elements.

References

- Neuberger, A., S. Peles, and D. Rittel, “Scaling the Response of Circular Plates Subjected to Large and Close-Range Spherical Explosions. Part I: Air-Blast Loading,” International Journal of Impact Engineering, vol. 34, pp. 859–873, 2007.

Model change

This section tests the removal and introduction of elements or contact pairs during the course of an analysis. The problems in this section can be divided into two groups of tests. The first group focuses on a simple uniaxial deformation mode and reintroduction of elements, without strain, in an annealed state. These tests are divided into sections according to the elements that can use the annealed element pair removal capability and by the analysis procedure used in the test. The second group is more general and focuses on the reintroduction of elements both with and without strain and with initial conditions. These tests are divided primarily into sections according to element type but include a number of miscellaneous tests.
The group of more general tests is described in *Stress/displacement model change: general tests*.

In this section:

- *Stress/displacement model change: static*
- *Stress/displacement model change: dynamic*
- *Stress/displacement model change: general tests*
- *Heat transfer model change: steady state*
- *Coupled temperature-displacement model change: steady state*
- *Contact model change*
- *Acoustic model change: steady state*
- *Pore-thermal model change*

Stress/displacement model change: static

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D8 C3D8I C3D8R
CAX4H CGAX3HT CGAX4HT CGAX4RH
CGAX6M CGAX6MH CGAX8HT CGAX8RHT
CPE4 CPE4I CPE4R CPE4RT CPE4RHT CPE8 CPS4 CPS4R CPS4RT
CPEG4RT CPEG4RHT CPEG6M CPEG6MH
DCOUP2D S4 SC8R SC6R T2D2 CSS8

Features tested

Continuum stress/displacement elements are removed and added during a static analysis. General nonlinear and linear perturbation steps are tested with elastic, hyperelastic, and plastic material properties. Various modeling features, such as multi-point constraints and transformed nodal and element variables, are tested in conjunction with element removal/reactivation.

Problem description

Model: All models have dimensions 5.0×2.0 in the x–y plane, with an out-of-plane dimension of 1.0 (plane stress/strain analysis). The axisymmetric models are 5.0 units in the z-direction and have an inner radius of 1.0 units.

Material: The material is assumed to be a compressible rubber, except in the elastic-plastic test. The material constants are not given in any specific set of units. The rubber is modeled both as a hyperelastic material and as a linear elastic material that matches the hyperelastic material at small strain.

Elastic material:

Young's modulus = 4.064385×10^6

Poisson's ratio = 0.451566

Hyperelastic material:

$C_{10} = 56.00 \times 10^4$

$C_{01} = 14.00 \times 10^4$

$D_1 = 1.43 \times 10^{-7}$

Elastic-plastic material:

Young's modulus = 3.0×10^6

Poisson's ratio = 0.3

Plastic hardening:

Yield stress	Plastic strain
0.15×10^5	0.0000
0.60×10^5	2.027×10^{-4}

Loading and boundary conditions:

General tests:

The loading in Step 1 is to compress the right-hand side of the model 0.1 units in the x-direction, while the left-hand side is fixed in the x-direction. In Step 2 the middle portion of the model, consisting of elements 2–4 and 7–9, is removed (see *Figure 1*). This releases the load in the remaining elements. In Step 3 the nodes of the removed elements are repositioned to their original positions in the y- and, if applicable, z-directions. In Step 4 the elements are added back into the model and the right-hand side nodes are displaced to the position $x = 5.1$, corresponding to a displacement of 0.1 units. The loading for the axisymmetric models is in the z-direction.

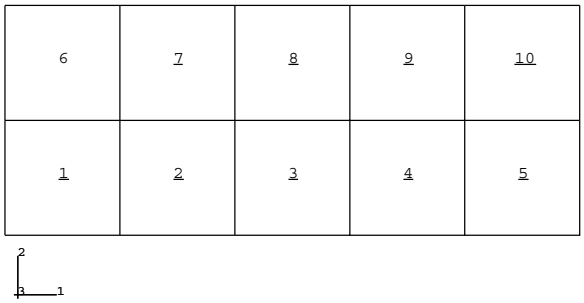


Figure 1: Mesh used in general static tests.

Specific tests:

The loading for the specific tests is identical to that used in the general tests with the following exceptions: *pmce_cpe8_se1.inp* and *pmce_cpe8_sh1.inp*, which have body loads active during all steps of the analysis; *pmce_c3d8_se1.inp*, which has a predefined temperature load but no displacement boundary condition (except to constrain rigid body motion); *pmce_cpe4i_se1.inp*, in which prescribed displacements are 10^{-2} times those of the other tests; and *pmce_cpe4_sp.inp* and *pmce_cpe4_sp1.inp*, where the displacement in the fourth step is such that only the newly introduced elements yield.

pmce_c3d8_se1.inp

The initial temperature is $\theta = 20$. The middle portion of the model is removed in Step 1. In Step 2 the temperature at the nodes of the removed elements is reset to $\theta = 100$. In Step 3 the nodal temperatures of the removed elements are set to $\theta = 180$, and the temperatures at the other nodes in

the model are reset to $\theta = 60$. The coefficient of thermal expansion for the middle elements is one half that of the other elements.

pmce_cpe4_sp.inp

In Step 4 the right-hand-side nodes are given an x-displacement of $u_x = -0.005$ so that only the reactivated elements yield in this step (having been annealed, they have not hardened as the other elements have).

pmce_cpe4_sp1.inp

This problem is identical to *pmce_cpe4_sp.inp*, except that a DCOUP2D element is removed in Step 1 and added in Step 4 to apply the x-displacement of $u_x = -0.005$.

pmce_cpe4i_se1.inp

The elements have a stiffness 100 times that of the elements in the other tests. Each of the elements has a single line of rebar that runs through the middle of the element parallel to the x-axis. The rebar has 1% of the cross-sectional area of the element at the face it cuts. It is given a stiffness in plane stress that is 100 times the plane strain modulus of the element. This ensures that the rebar exactly doubles the stiffness of the element. This model is verified with small displacements to avoid the effect of thinning of the rebar cross-section as it stretches.

pmce_cpe8_se1.inp and pmce_cpe8_sh1.inp

The body load on each of these models is equal to 70000 units in the x-direction.

Reference solution

These models (except for *pmce_cps4_se1.inp*, *pmce_cps4r_se1.inp*, *pmce_cgax4rh_se1.inp*, and *pmce_c3d8r_se1.inp*) include nonlinear geometric terms. This loading regime puts the model in uniaxial stress in the plane stress, axisymmetric, and three-dimensional models; in the plane strain model it is in a biaxial state of stress. Hence, the stress can be found by multiplying the strain by the elastic modulus or, in the case of plane strain, by $E' = E/\left(1-\nu^2\right)$. Only the strain value will be listed.

General tests:

Step 1

Uniform axial strain should exist in this step for all tests. The value should be ln (l/L_o), where l = 4.9 and L_o = 5.0. These values give $\epsilon_{xx} = -2.0203 \times 10^{-2}$.

Step 2

The stress and strain in the elements that are not removed should become zero. The nodes on elements 1 and 6 should have $u_x = 0.0$, and the nodes on elements 5 and 10 should have $u_x = -0.1$.

Step 3

The displacement of the nodes in this step should have no effect on the results that were obtained in Step 2.

Step 4

For the plane strain, axisymmetric, and three-dimensional models there will be a state of uniform axial strain in this step. The magnitude will be $\epsilon_{xx} = 4.0005 \times 10^{-2}$ (ln (l/L_o), where $l = 5.1$ and $L_o = 4.9$).

For the plane stress and truss elements there is a change in thickness of the elements in Step 1. The thickness is not changed when elements are removed. Therefore, the elements added back into the model in this step will not have the same axial stiffness (and, hence, axial strain) as the elements that were not removed. The variation in ϵ_{xx} is as follows: elements 1, 5, 6, and 10 have $\epsilon_{xx} = 4.07 \times 10^{-2}$; elements 2, 4, 7, and 9 have $\epsilon_{xx} = 3.92 \times 10^{-2}$; elements 3 and 8 have $\epsilon_{xx} = 3.99 \times 10^{-2}$.

The axisymmetric models are loaded in the z-direction.

Specific tests:

The models that have the same loading as the general tests have the same analytical solution.

pmce_cps4_se1.inp

Because this is a test without NLGEOM, the strain is always based on the change in displacement divided by the original length. This produces $\epsilon_{xx} = -2 \times 10^{-2}$ in Step 1 and 4×10^{-2} in Step 4.

pmce_c3d8_se1.inp

There should be zero response in the model in Step 1 and Step 2. In Step 3 there should be thermal strains in the model equal to $\alpha\left(\theta - \theta^I\right) = 80\alpha$ for the middle elements and 40α for the other elements. (These thermal strains are the same value since the α value for the middle elements is one-half of that for the other elements.) There should be no elastic strain in the model and no stress.

pmce_cpe4_sp.inp and pmce_cpe4_sp1.inp

In Step 1 the model will yield uniformly. $\epsilon_{xx}^{pl} = -1.879 \times 10^{-4}$. In Step 4 only the middle elements of the model will yield. ϵ_{xx}^{pl} will be approximately -1.76×10^{-4}.

pmce_cpe4i_se1.inp

The strains in Steps 1 and 4 are $\epsilon_{xx} = -2 \times 10^{-4}$ and 4×10^{-4}, respectively. This applies to both the rebar and the elements.

pmce_cpe8_se1.inp and pmce_cpe8_sh1.inp

In Step 1 there will be a gradient of ϵ_{xx} in the model. In Step 2 elements 1 and 6 will be in tension and Elements 5 and 10 will be in compression. In Step 4 there will be a gradient of ϵ_{xx} in the model.

Results and discussion

All models produce results that match the expected theoretical values.

Input files

General tests

pmce_c3d8i_se.inp	C3D8I elements, elastic material.
pmce_c3d8i_sh.inp	C3D8I elements, hyperelastic material.
pmce_c3d8r_se.inp	C3D8R elements, elastic material.
pmce_cax4h_se.inp	CAX4H elements, elastic material.
pmce_cax4h_sh.inp	CAX4H elements, hyperelastic material.
pmce_cgax3ht_sh.inp	CGAX3HT elements, hyperelastic material.
pmce_cgax4ht_sh.inp	CGAX4HT elements, hyperelastic material.
pmce_cgax4rh_sh.inp	CGAX4RH elements, hyperelastic material.
pmce_cgax6m_sh.inp	CGAX6M elements, hyperelastic material.
pmce_cgax6mh_sh.inp	CGAX6MH elements, hyperelastic material.
pmce_cgax8ht_sh.inp	CGAX8HT elements, hyperelastic material.
pmce_cgax8rht_sh.inp	CGAX8RHT elements, hyperelastic material.
pmce_cpe4r_sh.inp	CPE4R elements, hyperelastic material.
pmce_cpe4rt_se.inp	CPE4RT elements, elastic material.
pmce_cpe4rt_sh.inp	CPE4RT elements, hyperelastic material.
pmce_cpe4rht_se.inp	CPE4RHT elements, elastic material.
pmce_cpe4rht_sh.inp	CPE4RHT elements, hyperelastic material.
pmce_cpeg4rt_se.inp	CPEG4RT elements, elastic material.
pmce_cpeg4rt_sh.inp	CPEG4RT elements, hyperelastic material.
pmce_cpeg4rht_se.inp	CPEG4RHT elements, elastic material.
pmce_cpeg4rht_sh.inp	CPEG4RHT elements, hyperelastic material.
pmce_cpe8_se.inp	CPE8 elements, elastic material.
pmce_cpe8_sh.inp	CPE8 elements, hyperelastic material.
pmce_cps4_se.inp	CPS4 elements, elastic material.
pmce_cps4_sh.inp	CPS4 elements, hyperelastic material.
pmce_cps4rt_se.inp	CPS4RT elements, elastic material.
pmce_cps4rt_sh.inp	CPS4RT elements, hyperelastic material.
pmce_cpeg6m_sh.inp	CPEG6M elements, hyperelastic material.
pmce_cpeg6mh_sh.inp	CPEG6MH elements, hyperelastic material.
pmce_s4_se.inp	S4 elements, elastic material.
pmce_sc8r_se.inp	SC8R elements, elastic material.
pmce_sc6r_se.inp	SC6R elements, elastic material.
pmce_t2d2_se.inp	T2D2 elements, elastic material.
pmce_css8_se.inp	CSS8 elements, elastic material.
pmce_css8_sh.inp	CSS8 elements, hyperelastic material.

Specific tests

pmce_c3d8_se1.inp	C3D8 elements with **TEMPERATURE*.
pmce_c3d8i_sh1.inp	C3D8I elements with **TRANSFORM* on all nodes.
pmce_c3d8r_se1.inp	C3D8R elements without NLGEOM.
pmce_cgax4rh_se1.inp	CGAX4RH elements without NLGEOM.
pmce_cpe4_se1.inp	CPE4 elements with **ORIENTATION*.
pmce_cpe4_sh1.inp	CPE4 elements with *UHYPER* and perturbation step.
pmce_cpe4_sh1.f	User subroutine *UHYPER* used in pmce_cpe4_sh1.inp.
pmce_cpe4_sp.inp	CPE4 elements with elastic-plastic material.
pmce_cpe4_sp1.inp	CPE4 and DCOUP2D elements with elastic-plastic material.
pmce_cpe4i_se1.inp	CPE4I elements with rebar.
pmce_cpe8_se1.inp	CPE8 elements with GRAV-type **DLOAD*.
pmce_cpe8_sh1.inp	CPE8 elements with BX-type **DLOAD*.
pmce_cps4_se1.inp	CPS4 elements without NLGEOM.
pmce_cps4_sh1.inp	CPS4 elements with **MPC*.
pmce_cps4r_se1.inp	CPS4R elements without NLGEOM.

Stress/displacement model change: dynamic

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

CPS4

Features tested

We examine how the natural frequencies of two different systems change when their mass and geometry change with element removal. The removed elements are also added back so that the response of the original system is recovered. The dominant mode is computed with a frequency extraction run, as well as by using direct-integration dynamics.

Problem description

The frequency extraction and dynamic analyses are discussed below.

Frequency analysis

A natural frequency extraction is carried out on a cantilevered beam. Elements are removed to shorten the length of the beam, thereby changing the frequency content.

Material properties:

Hyperelastic material, polynomial, N=1

$C_{10} = 56.0 \times 10^4$, $C_{01} = 14.0 \times 10^4$, $D_1 = 1.43 \times 10^{-7}$

Dimensions:

10.0×1.0 in the x–y plane, 1.0 out-of-plane.

Loading and boundary conditions:

The beam is cantilevered at one end. Step 1 is a null step to establish the base state. In Step 2 the first five eigenvalues are extracted. In Step 3 elements 4–6,14–16, 24–26, and 34–36 are removed. In Step 4 the end of the beam is shortened by 0.5 units ($u_x = -0.5$). The removed elements are added back into the model in Step 5. The first five eigenvalues are obtained for the shortened beam in Step 6. *Figure 1* shows the mesh used in this test.

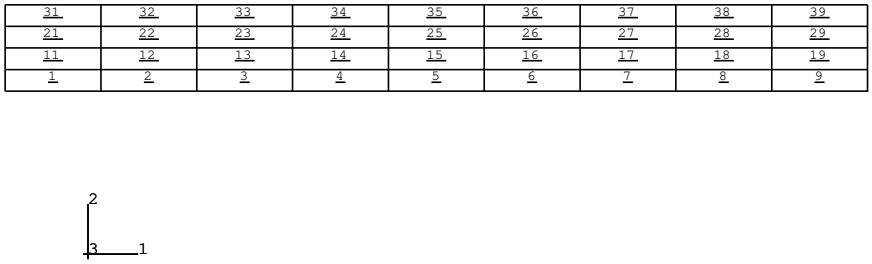


Figure 1: Mesh used in frequency procedure test.

Dynamic analysis

A block of eight elements attached to a grounded spring is given an initial displacement out of static equilibrium and is allowed to vibrate. The response is compared to that of the same system vibrating with one-quarter of the original mass.

Material properties:

Elastic modulus = 207.0×10^{12}

Poisson's ratio = 0.3

Density = 7800.0

Spring stiffness = 9.8538×10^{6}

Dimensions:

The models have dimensions 8.0×4.0 in the x–y plane, 1.0 out-of-plane.

Loading and boundary conditions:

The nodes at the right-hand side of the model are displaced by 1.0 units along the 1-direction in Step 1. In Step 2 they are released. All nodes are constrained to slide along the 1-direction only. In Step 3 all of the nodes are held fixed and elements 2–4 and 12–14 are removed. The remaining elements are allowed to move in Step 4. During this step the free nodes (i.e., no mass contributed by any elements) on the removed elements are held fixed. In Step 5 the entire model is again held fixed and the elements are added back into the model. In Step 6 all of the nodes are released. The mesh for this test is shown in *Figure 2*.

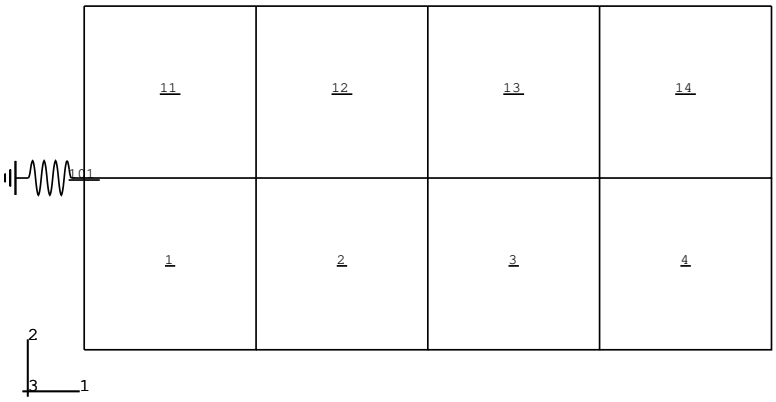


Figure 2: Mesh used in dynamic procedure test.

Reference solution

The natural frequencies for a cantilever beam are given by

$$\lambda = c^2 \sqrt{\frac{EI}{\rho A}}$$

in rad s^{-1}. For the first mode $c = 1.875/L$, where L is the beam length.

The natural frequency for the spring-mass system is given by $\sqrt{k/m}$, where k is the spring stiffness and m is the total mass of the block.

Results and discussion

The first natural frequency of the cantilever beam was found to be within 2% of the analytical solution. The period for the spring-mass system in transient dynamics matches the expected analytical solution shown above for all of the dynamic steps (Steps 2, 4, and 6). The total force on the vertical left edge is output.

Input files

pmce_cps4_f.inp	CPS4 elements in a frequency extraction analysis.
pmce_cps4_d.inp	CPS4 elements in a dynamic analysis.

Stress/displacement model change: general tests

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

Structural elements

B21 B22 B23 B21H B22H B23H B31 B32 B33 B31H B31OS B31OSH B32H B32OS B32OSH PIPE21 PIPE21H PIPE22 PIPE22H PIPE31 PIPE31H PIPE32 PIPE32H ELBOW31 ELBOW31B ELBOW31C ELBOW32 MAX1 MAX2 MGAX1 MGAX2 M3D3 M3D4 M3D4R M3D8 M3D8R M3D9R SAX1 SAX2 SAXA11 SAXA21 STRI3 S3 S3R S4R STRI65 S4 S4R5 S8R S8R5 S9R5

Two-dimensional continuum elements

CAX4 CAX4H CAX4I CAX4R CAX8 CAX8H CAX8R CGAX3HT CGAX3T CGAX4 CGAX4H CGAX4HT CGAX4R CGAX4RHT CGAX4RT CGAX4T CGAX8 CGAX8H CGAX8HT CGAX8R CGAX8RHT CGAX8RT CGAX8T CPEG4T CPEG8 CPEG8H CPEG8R CINAX4 CINPS4 CPE4 CPE4H CPE4I CPE4R CPE4RP CPE4RT CPE4T CPE8 CPE8H CPE8P CPE8PH CPE8R CPE8RP CPS3 CPS4 CPS4I CPS4R CPS4RT CPS4T CPS6 CPS6M CPS8 CPS8R

Three-dimensional continuum elements

C3D4 C3D4H C3D6 C3D6H C3D8 C3D8H C3D8I C3D8IH C3D8R C3D8RH C3D8RT C3D10 C3D10H C3D10MH C3D10MT C3D15 C3D15V C3D15VH C3D20 C3D20H C3D20R C3D20H C3D20RH C3D27 C3D27H C3D27R C3D27H C3D27RH CIN3D8

Miscellaneous elements

DASHPOT1 DASHPOT2 JOINTC LS3S LS6 MASS SPRING1 SPRING2

Features tested

This section includes a very general set of tests for the reactivation with strain of stress/displacement elements.

Each test contains a pair of bodies, each modeled with either one or two elements, so the tests are one-element or two-element tests. In many cases more than one pair of bodies is in a single input file.

One of these bodies, the reference body, is loaded in various ways without ever being removed from the analysis. The other body, the test body, has the same material and thickness properties; however, the test body has a significantly different initial configuration than the reference body, in such a way that it has different stiffness, volume, and mass. During the first step of the analysis, the test body is deformed into the same shape as the reference body. It is then removed and reactivated strain free while in this configuration so that the initial configurations of the two bodies are now identical.

The bodies are then given identical loadings, and the behavior of the two bodies should be identical. To test reactivation with strain, a further removal and a reactivation, this time with strain, occurs for the test body. This kind of reactivation does not reset the initial configuration of the test body, so the behavior of the two bodies should still be identical.

Problem description

Outline of steps (Steps 8–13 are applied for all but the few element types for which dynamic steps are not supported):

1. Deform the test body into the identical shape as the undeformed reference body. For structural elements this requires applying rotations as well as displacements at the nodes of the test elements so that the normals, as well as the nodal coordinates, coincide.
2. Remove the test body.
3. Reactivate the test body without strain. The reactivation should reactivate the test body in the identical configuration as the reference body. Element properties such as cross-sectional area or thickness will be reset to their values at the beginning of the analysis.
4. Apply element loading to both bodies. Both bodies will deform in the same manner. The load applied during this step will remain active throughout the remainder of the analysis.
5. Remove the test body. Since the applied loads are element loads, they will be removed from the test body automatically.
6. Reactivate the test body with strain. Previously applied loads will be reactivated automatically as well. No additional loads are applied during this step. The final configuration of the test body will be identical to that at the end of Step 4 for both bodies. From this step through the remainder of the analysis, the test and reference bodies will provide identical results.
7. Apply a thermal load to both bodies.
8. Perform a frequency extraction. Eigenvalues occur in pairs because of the pairs of identical bodies in the input file. Sufficient eigenvectors must be extracted to represent each body of each pair of bodies equally.
9. Perform a transient modal dynamic analysis using the extracted eigenmodes. A concentrated load is applied at one node. The duration of the analysis is approximately one-tenth of the first fundamental time period.
10. Perform a mode-based steady-state dynamic analysis. The frequency sweep is performed approximately up to the first 10 natural frequencies.
11. Perform a direct-solution steady-state dynamic analysis. The frequency sweep is performed approximately up to the first 10 natural frequencies.
12. Perform a subspace-based steady-state dynamic analysis. The frequency sweep is performed approximately up to the first 10 natural frequencies.
13. Perform a nonlinear transient dynamic analysis using the direct-integration dynamic implict procedure. The loads and time step size used are those used in Step 9.

A second set of verification problems is added to test the element loads. These are the input file names with _dl added to the end of the file name. The first four steps are identical to the test described above. The remainder of the steps test most of the available distributed load options for each element type.

A final category of tests includes material and initial conditions tests. This group of verification also consists of two-body test cases. These analyses apply initial conditions to various material properties such as void ratio, kinematic shift tensor, and others.

The test body is removed during the first step of the analyses. Because removal occurs in the first step, the initial conditions will remain in place when the test body is reintroduced strain free in the second step. Displacement boundary conditions are then applied to both bodies, which must show identical behavior.

Results and discussion

It is not necessary to check the results to an analytical solution for these tests. However, it is necessary to determine if the test body is being reintroduced back into the analysis properly. Proper reintroduction requires that the test and reference bodies behave identically after the second step. All test elements produce results that match the reference elements.

Input files

Structural element tests

pmcp_beam2d.inp	Two-dimensional beam elements.
pmcp_beam2d_dl.inp	Two-dimensional beam elements, element loading test.
pmcp_beam3d.inp	Three-dimensional beam elements.
pmcp_beam3d_dl.inp	Three-dimensional beam elements, element loading test.
pmcp_pipe2d.inp	Two-dimensional pipe elements.
pmcp_pipe2d_dl.inp	Two-dimensional pipe elements, element loading tests.
pmcp_pipe3d.inp	Three-dimensional pipe elements.
pmcp_pipe3d_dl.inp	Three-dimensional pipe elements, element loading tests.
pmcp_elbow3d.inp	Elbow elements.
pmcp_shell.inp	General shell elements; includes rebar, cylindrical orientations, and **NODAL THICKNESS*.
pmcp_shell_dl.inp	General shell elements, element loading tests.
pmcp_shell5.inp	5-degree-of-freedom general shell elements; includes rebar, cylindrical orientations, and **NODAL THICKNESS*; no reactivation WITH STRAIN.
pmcp_shell5_dl.inp	5-degree-of-freedom general shell elements, element loading tests.
pmcp_memb.inp	General membrane elements; includes rebar, orientations, and **NODAL THICKNESS*.
pmcp_memb_dl.inp	General membrane elements, element loading tests.
pmcp_aximemb.inp	Axisymmetric membrane elements, element loading tests; includes rebar.
pmcp_axishell.inp	Axisymmetric shell elements, element loading tests; includes rebar.
pmcp_saxa.inp	Axisymmetric shell elements with nonaxisymmetric loading, excluding element loading tests; includes rebar.

Two-dimensional continuum element tests

pmcp_cax.inp	Axisymmetric continuum elements.
pmcp_cax_dl.inp	Axisymmetric continuum elements, load test.
pmcp_cgax.inp	Axisymmetric elements with twist; includes load test, excludes dynamic steps.
pmcp_cgax3t.inp	Coupled temperature-displacement axisymmetric triangular elements with twist; includes load test, excludes dynamic steps.
pmcp_cgaxt.inp	Coupled temperature-displacement axisymmetric quadrilateral elements with twist; includes load test, excludes dynamic steps.
pmcp_cpe.inp	Plane strain elements.

pmcp_cpe_res.inp	Restart from pmcp_cpe.inp; Step 4 onward duplicated.
pmcp_cpe4_ori.inp	CPE4 element with **ORIENTATION*.
pmcp_cpe4h_ori.inp	CPE4H element with **ORIENTATION*.
pmcp_cpe8_ori.inp	CPE8 element with **ORIENTATION*.
pmcp_cpeg.inp	Generalized plane strain elements.
pmcp_cps.inp	Plane stress elements.
pmcp_cpstri.inp	Triangular plane stress elements.
pmcp_cps4_ori.inp	CPS4 element with **ORIENTATION*.
pmcp_cps8_ori.inp	CPS8 element with **ORIENTATION*.
pmcp_infinite2d.inp	Infinite elements.
pmcp_2dpore.inp	Pore pressure plane strain elements.
pmcp_2drebar.inp	Plane strain, plane stress, and axisymmetric elements with rebar.
pmcp_2drebar_res.inp	Restart from pmcp_2drebar.inp; Step 4 onward duplicated.
pmcp_2dtemp.inp	Coupled temperature-displacement elements.
pmcp_2dtemp_cgaxt.inp	Coupled temperature-displacement axisymmetric quadrilateral elements with twist.
pmcp_2dtemp_cgax3t.inp	Coupled temperature-displacement axisymmetric triangular elements with twist.
pmcp_2dtemp_cpeg4t.inp	Coupled temperature-displacement generalized plane strain quadrilateral elements.
pmcp_2dtemp_nodyn.inp	Coupled temperature-displacement generalized plane strain quadrilateral elements, excludes dynamic steps.

Three-dimensional continuum element tests

pmcp_c3d8.inp	C3D8 element with **ORIENTATION* and **REBAR*.
pmcp_c3d8_dl.inp	C3D8 element loading test.
pmcp_c3d8h.inp	General test of C3D8H element.
pmcp_c3d8h_dl.inp	C3D8H element loading test.
pmcp_c3d8i.inp	General test of C3D8I element.
pmcp_c3d8i_dl.inp	C3D8I element loading test.
pmcp_c3d8ih.inp	General test of C3D8IH element.
pmcp_c3d8ih_dl.inp	C3D8IH element loading test.
pmcp_c3d8r.inp	General test of C3D8R element.
pmcp_c3d8r_dl.inp	C3D8R element loading test.
pmcp_c3d8rh.inp	General test of C3D8RH element.
pmcp_c3d8rh_dl.inp	C3D8RH element loading test.
pmcp_c3d8rt.inp	General test of C3D8RT element.
pmcp_c3d8rt_dl.inp	C3D8RT element loading test.
pmcp_c3d10mt.inp	General test of C3D10MT element.
pmcp_c3d10mt_dl.inp	C3D10MT element loading test.
pmcp_c3d15v.inp	General test of C3D15V element.
pmcp_c3d15v_dl.inp	C3D15V element loading test.
pmcp_c3d15vh.inp	General test of C3D15VH element.
pmcp_c3d15vh_dl.inp	C3D15VH element loading test.

pmcp_c3d20.inp	General test of C3D20 element.
pmcp_c3d20_dl.inp	C3D20 element loading test.
pmcp_c3d20h.inp	General test of C3D20H element.
pmcp_c3d20h_dl.inp	C3D20H element loading test.
pmcp_c3d20r.inp	General test of C3D20R element.
pmcp_c3d20r_dl.inp	C3D20R element loading test.
pmcp_c3d20rh.inp	General test of C3D20RH element.
pmcp_c3d20rh_dl.inp	C3D20RH element loading test.
pmcp_c3d27.inp	General test of C3D27 element.
pmcp_c3d27_dl.inp	C3D27 element loading test.
pmcp_c3d27h.inp	General test of C3D27H element.
pmcp_c3d27h_dl.inp	C3D27H element loading test.
pmcp_c3d27r.inp	General test of C3D27R element.
pmcp_c3d27r_dl.inp	C3D27R element loading test.
pmcp_c3d27rh.inp	General test of C3D27RH element.
pmcp_c3d27rh_dl.inp	C3D27RH element loading test.
pmcp_infinite3d.inp	CIN3D8, infinite element test.
pmcp_prisms.inp	General test of C3D6, C3D6H, C3D15, and C3D15H elements.
pmcp_prisms_dl.inp	Element loading test of C3D6, C3D6H, C3D15, and C3D15H elements.
pmcp_tets.inp	General test of tetrahedral elements.
pmcp_tets_dl.inp	Element loading test of tetrahedral elements.

Miscellaneous element tests

pmcp_jointc.inp	JOINTC element.
pmcp_linespring.inp	Line spring element.
pmcp_spring.inp	Spring, dashpot, and mass elements.
pmcp_substr.inp	Substructures constructed of beam elements.

Material/initial condition tests

pmcp_cap.inp	Cap plasticity with initial stresses.
pmcp_clay.inp	Cam-clay model with initial stresses.
pmcp_foam.inp	Crushable foam material.
pmcp_hyperelastic.inp	Hyperelastic material.
pmcp_hyperfoam.inp	Hyperfoam material.
pmcp_initstress.inp	Initial stresses in elements and **NODAL THICKNESS* are included.
pmcp_mises.inp	Mises plasticity with initial shift in kinematic hardening tensor.
pmcp_porous.inp	Porous metal plasticity.

These tests include elements CAX4, CAXA41, CPE4, CPS4, CGAX4, S4R, and M3D4R. Rebars are included with the CAX4, M3D4R, and S4R elements.

Heat transfer model change: steady state

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

DC2D4 DS3

Features tested

Continuum and shell heat transfer elements are removed and added during a steady-state heat transfer analysis.

Problem description

Model: The models have dimensions 5.0 × 2.0 in the *x*–*y* plane, with an out-of-plane dimension of 1.0.

Material:

Conductivity	7.872×10^{-4}
Density	0.2829

Loading and boundary conditions: The left side of the model is held at $\theta^0 = 0.0$. There is a film condition on the right side of the model for the simulation with DC2D4 elements and a temperature boundary condition for the simulation with DS3 shell elements. The sink temperature is $\theta^0 = 100.0$, and the film coefficient, h, is 1.0. For the shell problem the temperature boundary condition is 100.0 on the right-hand edge. A steady-state solution is obtained. Then two-thirds of the model is removed. When the elements are removed, the temperatures along the new external boundary are held fixed. The removed elements are added back into the model in the last step, and a new film condition is applied on the right-hand side for the continuum model and a temperature boundary condition for the shell model. The new sink temperature is $\theta^0 = 200.0$, and the same film coefficient is used. The temperature boundary condition is 200.0.

Reference solution

The one-dimensional heat balance equation for steady-state heat transfer is

$$-k\frac{d\theta}{dx} = q.$$

This equation can be integrated to give $\theta(x)$. Using the boundary conditions that $\theta = 0.0$ at $x = 0$ and that $q = h\left(\theta - \theta^0\right)$ at $x = L$, the solution for the continuum model is

$$\theta(x) = \frac{x}{L + k/h}\,\theta^0.$$

This expression can be used to calculate the temperature distribution in the model for the first and third steps. For the shell model the boundary conditions and the integration yield a linear temperature profile along the length of the model.

Results and discussion

The model gives the theoretical results in both the first and third steps.

Input files

pmce_dc2d4_h.inp	General test of DC2D4 elements in steady-state analysis.
pmce_ds3_h.inp	Test of element type DS3 in steady-state analysis.

Coupled temperature-displacement model change: steady state

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D8HT C3D8RT C3D8RHT C3D8T C3D10MT C3D10MHT CAX4RHT CAX4RT CAX6MHT CAX6MT CGAX3HT CGAX3T CGAX4HT CGAX4RT CGAX4RHT CGAX4T CGAX6MHT CGAX6MT CGAX8HT CGAX8RT CGAX8RHT CGAX8T CPE4RT CPE4RHT CPE4T CPE6MHT CPE6MT CPEG3HT CPEG3T CPEG4HT CPEG4RT CPEG4RHT CPEG4T CPEG6MHT CPEG6MT CPEG8HT CPEG8RHT CPEG8T CPS4RT CPS4T CPS6MT CPS8RT CPS8T

Features tested

Continuum coupled temperature-displacement elements are removed and added during a steady-state analysis.

Problem description

Model: The models have dimensions 5.0 × 2.0 in the *x*–*y* plane with an out-of-plane dimension of 1.0. In the axisymmetric case the models have dimensions 2.0 × 5.0 in the *r*–*z* plane, and the inner radius, R_i, equals 10^5. The inner radius is large to ensure that the strains in the circumferential direction are approximately uniform, which allows a comparison of the results obtained in this analysis with those obtained analytically.

Material:

Conductivity	7.872×10^{-4}
Density	0.2829
Thermal expansion	1.0×10^{-6}
Elastic modulus	100×10^4
Poisson's ratio	0.25

Loading and boundary conditions: The left side of the model is held at θ=0.0. There is a film condition on the right side of the model. The sink temperature is θ^0=100.0, and the film coefficient, *h*, is 1.0. After a steady-state solution is obtained, some of the elements in the model are removed. The temperatures along the new external boundary are held fixed. The removed elements are added back into the model in the last step, and a new film condition is applied on the right side. The new sink temperature is θ^0=200.0, and the same *h* is used.

During all three steps the following mechanical boundary conditions are maintained: u_y=0.0 at all points along *y*=0; u_x=0.0 at the point (0,0).

Reference solution

The solution for the one-dimensional steady-state heat transfer problem is given in *Heat transfer model change: steady state*. The solution for the mechanical response of the model is

$$\epsilon_x = \frac{\partial u}{\partial x} = \epsilon_x^{the} = \alpha\theta(x) = \alpha \frac{x}{L+k/h} \theta^0.$$

The expression for ϵ_x is integrated to give

$$u = \frac{1}{2}\alpha \frac{x^2}{L+k/h} \theta^0 + f(y).$$

The *y*-component of strain is given as

$$\epsilon_y = \frac{\partial v}{\partial y} = \alpha \frac{x}{L+k/h} \theta^0.$$

Integrating for *v* gives

$$v = \alpha \frac{xy}{L+k/h} \theta^0,$$

where the boundary condition that *v*=0 at *y*=0 is used to eliminate the terms that are only functions of *x*. The condition that

$$\gamma_{xy} = \frac{\partial u}{\partial y} + \frac{\partial v}{\partial x} = 0$$

is used to find $f(y)$, and the *x*-displacement is given as

$$u = \frac{\alpha}{2} \frac{(x^2-y^2)}{L+k/h} \theta^0.$$

These expressions are used to calculate the displacements in the model. The temperature distribution can be calculated with the expression from *Heat transfer model change: steady state*. The results for the axisymmetric case are obtained by replacing *x* with *z* and *y* with $(r - R_i)$ in the relations for temperature and displacements. In addition, the displacements are multiplied by a factor of $(1 + \nu)$, where ν is the Poisson's ratio. This takes into account the contribution from the approximately constant strain in the circumferential direction.

Results and discussion

The model produces the theoretical results in both the first and third steps for the element temperatures and for the quadratic element displacements. The displacements obtained using the model with linear elements do not match the theoretical results but are still reasonable.

Input files

pmce_c3d8ht_ctd.inp	General test of C3D8HT elements in steady-state analysis.
pmce_c3d8rht_ctd.inp	General test of C3D8RHT elements in steady-state analysis.
pmce_c3d8rt_ctd.inp	General test of C3D8RT elements in steady-state analysis.
pmce_c3d8t_ctd.inp	General test of C3D8T elements in steady-state analysis.
pmce_c3d10mht_ctd.inp	General test of C3D10MHT elements in steady-state analysis.
pmce_c3d10mt_ctd.inp	General test of C3D10MT elements in steady-state analysis.
pmce_cax4rht_ctd.inp	General test of CAX4RHT elements in steady-state analysis.
pmce_cax4rt_ctd.inp	General test of CAX4RT elements in steady-state analysis.

pmce_cax6mht_ctd.inp	General test of CAX6MHT elements in steady-state analysis.
pmce_cax6mt_ctd.inp	General test of CAX6MT elements in steady-state analysis.
pmce_cgax3ht_ctd.inp	General test of CGAX3HT elements in steady-state analysis.
pmce_cgax3t_ctd.inp	General test of CGAX3T elements in steady-state analysis.
pmce_cgax4ht_ctd.inp	General test of CGAX4HT elements in steady-state analysis.
pmce_cgax4rht_ctd.inp	General test of CGAX4RHT elements in steady-state analysis.
pmce_cgax4rt_ctd.inp	General test of CGAX4RT elements in steady-state analysis.
pmce_cgax4t_ctd.inp	General test of CGAX4T elements in steady-state analysis.
pmce_cgax6mht_ctd.inp	General test of CGAX6MHT elements in steady-state analysis.
pmce_cgax6mt_ctd.inp	General test of CGAX6MT elements in steady-state analysis.
pmce_cgax8ht_ctd.inp	General test of CGAX8HT elements in steady-state analysis.
pmce_cgax8rht_ctd.inp	General test of CGAX8RHT elements in steady-state analysis.
pmce_cgax8rt_ctd.inp	General test of CGAX8RT elements in steady-state analysis.
pmce_cgax8t_ctd.inp	General test of CGAX8T elements in steady-state analysis.
pmce_cpe4rht_ctd.inp	General test of CPE4RHT elements in steady-state analysis.
pmce_cpe4rt_ctd.inp	General test of CPE4RT elements in steady-state analysis.
pmce_cpe4t_ctd.inp	General test of CPE4T elements in steady-state analysis.
pmce_cpe6mht_ctd.inp	General test of CPE6MHT elements in steady-state analysis.
pmce_cpe6mt_ctd.inp	General test of CPE6MT elements in steady-state analysis.
pmce_cpeg3ht_ctd.inp	General test of CPEG3HT elements in steady-state analysis.
pmce_cpeg3t_ctd.inp	General test of CPEG3T elements in steady-state analysis.
pmce_cpeg4ht_ctd.inp	General test of CPEG4HT elements in steady-state analysis.
pmce_cpeg4rht_ctd.inp	General test of CPEG4RHT elements in steady-state analysis.
pmce_cpeg4rt_ctd.inp	General test of CPEG4RT elements in steady-state analysis.
pmce_cpeg4t_ctd.inp	General test of CPEG4T elements in steady-state analysis.
pmce_cpeg6mht_ctd.inp	General test of CPEG6MHT elements in steady-state analysis.
pmce_cpeg6mt_ctd.inp	General test of CPEG6MT elements in steady-state analysis.
pmce_cpeg8ht_ctd.inp	General test of CPEG8HT elements in steady-state analysis.
pmce_cpeg8rht_ctd.inp	General test of CPEG8RHT elements in steady-state analysis.
pmce_cpeg8t_ctd.inp	General test of CPEG8T elements in steady-state analysis.
pmce_cps4rt_ctd.inp	General test of CPS4RT elements in steady-state analysis.
pmce_cps4t_ctd.inp	General test of CPS4T elements in steady-state analysis.
pmce_cps6mt_ctd.inp	General test of CPS6MT elements in steady-state analysis.
pmce_cps8rt_ctd.inp	General test of CPS8RT elements in steady-state analysis.
pmce_cps8t_ctd.inp	General test of CPS8T elements in steady-state analysis.

Contact model change

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

ISL21A

Features tested

Contact pairs and special-purpose contact elements are removed and added during a static analysis.

Problem description

The analyses in this section simulate a block sliding over another block. The models are taken through the following steps:

- In Step 1 the contact surfaces are brought together, resulting in the development of contact pressure at the interface.
- In Step 2 the secondary surface slides over the main surface to generate friction forces. The sliding motion is applied with boundary conditions at the top nodes.
- In Step 3 the contact pair is removed. The contact constraint ends immediately, and throughout this step the secondary surface penetrates into the main surface as the contact forces are ramped down to zero. At the end of this step there is no stress in the model.
- The contact pair is reactivated again in Step 4 by specifying contact interference with an allowed overclosure value. This value has been specified to be equal to the actual overclosure. As this allowed overclosure value ramps down to zero, the contact surfaces come gradually into compliance throughout the step.

Results and discussion

The contact condition is observed to behave as described.

Input files

pmcc_2dls.inp	Removal and reactivation of a two-dimensional finite-sliding contact pair.
pmcc_3dls.inp	Removal and reactivation of a three-dimensional finite-sliding contact pair.
pmcc_isl21.inp	Removal and reactivation of an ISL21A element.

Acoustic model change: steady state

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

AC1D2 AC1D3
AC2D3 AC2D4 AC2D6 AC2D8
AC3D4 AC3D5 AC3D6 AC3D8 AC3D10 AC3D15 AC3D20
ACAX3 ACAX4 ACAX6 ACAX8
ACINAX2 ACINAX3 ACIN2D2 ACIN2D3 ACIN3D3 ACIN3D4 ACIN3D6 ACIN3D6

Features tested

Continuum acoustic elements are removed during a steady-state acoustic analysis.

Problem description

Model: The models have dimensions 10.0 × 1.0 in the *x*–*y* plane, with an out-of-plane dimension of 1.0.

Material:

Bulk modulus, K_f	1.42176×10^5
Density, ρ_f	1.293

Loading and boundary conditions: The pressure at the left side of the model is constrained to equal one; there is a plane-wave radiation condition on the right side of the model. A steady-state solution at unit frequency is obtained. Then, one-half of the model is removed. When the elements are removed, the radiation condition is applied along the new external boundary.

Reference solution

The one-dimensional Helmholtz equation for steady-state acoustics is

$$d^2p/dx^2 + k^2p = 0,$$

where $k = \sqrt{K_f/\rho_f}$ is the acoustic wave number. Using the boundary conditions that $p = 1.0$ at $x = 0$ and that $dp/dx = -ikp$ at $x = L$, the solution for the continuum model is

$$|p(x)| = 1,$$

while the phase is consistent with a sine wave at the specified frequency, 1 Hertz.

Results and discussion

The model gives the theoretical results in both the first and second steps.

Input files

pmcp_ac1d2.inp	General test of AC1D2 elements in steady-state analysis.
pmcp_ac1d3.inp	AC1D3 elements.
pmcp_ac2d3.inp	AC2D3 elements.
pmcp_ac2d4.inp	AC2D4 elements.
pmcp_ac2d6.inp	AC2D6 elements.
pmcp_ac2d8.inp	AC2D8 elements.
pmcp_ac3d4.inp	AC3D4 elements.
pmcp_ac3d5.inp	AC3D5 elements.
pmcp_ac3d6.inp	AC3D6 elements.
pmcp_ac3d8.inp	AC3D8 elements.
pmcp_ac3d10.inp	AC3D10 elements.
pmcp_ac3d15.inp	AC3D15 elements.
pmcp_ac3d20.inp	AC3D20 elements.
pmcp_acax3.inp	ACAX3 elements.
pmcp_acax4.inp	ACAX4 elements.
pmcp_acax6.inp	ACAX6 elements.
pmcp_acax8.inp	ACAX8 elements.
pmcp_acinax2.inp	ACINAX2 elements.
pmcp_acinax3.inp	ACINAX3 elements.
pmcp_acin2d2.inp	ACIN2D2 elements.
pmcp_acin2d3.inp	ACIN2D3 elements.
pmcp_acin3d3.inp	ACIN3D3 elements.
pmcp_acin3d4.inp	ACIN3D4 elements.
pmcp_acin3d6.inp	ACIN3D6 elements.
pmcp_acin3d8.inp	ACIN3D8 elements.

Pore-thermal model change

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D8PT C3D8RPT

Features tested

Coupled pore-thermal elements are removed.

Problem description

Model: The models have dimensions 40.0 × 40 in the *x*–*y* plane, with an out-of-plane dimension of 5.0.

Material:

Modulus	88.7×10^{9}
Density	1922
Expansion	5.6×10^{-6}
Specific heat	3000
Conductivity	3
Density, pore fluid	1000
Expansion, pore fluid	$69. \times 10^{-6}$
Specific heat, pore fluid	4186
Conductivity, pore fluid	0.58
Permeability	0.001
Specific weight of fluid	9800

Loading and boundary conditions: Pore pressure of 10 units and surface pressure of 2 units is applied on the top surface. The temperature at the top surface is set to 100 units. Normal displacement is constrained on three faces. Elements at the center of the block are removed and added in various steps.

Results and discussion

The models produce results that match the expected values.

Input files

mdlchng_c3d8pt_p.inp General test of C3D8PT elements.

mdlchng_c3d8rpt_p.inp	General test of C3D8RPT elements.
mdlchng_c3d8rpt_t.inp	General test of C3D8RPT elements.

Symmetric model generation and analysis of cyclic symmetry models

In this section:

- *Symmetric model generation and results transfer*
- *Analysis of cyclic symmetric models*

Symmetric model generation and results transfer

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Symmetric model generation by revolving the cross-section of an axisymmetric mesh about the symmetry axis

Elements tested

Continuum elements

CAX3 CAX3H CAX4 CAX4H CAX4I CAX4IH CAX4R CAX4RH CAX6 CAX6H CAX8 CAX8H CAX8R CGAX3 CGAX3H CGAX4 CGAX4H CGAX4R CGAX4RH CGAX6 CGAX6H CGAX8 CGAX8H CGAX8RH CAX4T CAX4RT CAX4HT CAX8T CAX8RT CAX8HT CAX8RHT CGAX4T CGAX8T DCAX4

Shell and membrane elements

SAX1 SAX2 MAX1 MAX2 MGAX1 MGAX2 DSAX1 DSAX2

Surface elements embedded in continuum elements

SFMAX1 SFMAX2 SFMGAX1 SFMGAX2

Features tested

Symmetric model generation by revolving the cross-section of an axisymmetric mesh about the symmetry axis and symmetric results transfer.

Problem description

Material:

Young's modulus	1×10^4
Poisson's ratio	0.3
C10 (hyperelastic, hybrids only)	1.9×10^3
D11 (hyperelastic, hybrids only)	2.4×10^{-4}
Young's modulus (rebars)	1×10^6
Poisson's ratio (rebars)	0.3

Loading and boundary conditions: The loading and boundary conditions on the axisymmetric continuum element model are depicted in *Figure 1*. The loading and boundary conditions on the axisymmetric shell and membrane element model are depicted in *Figure 2*.

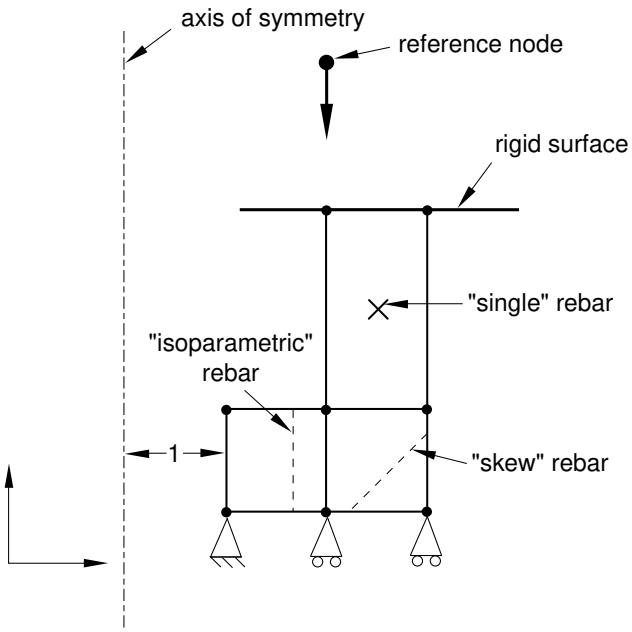


Figure 1: Axisymmetric model with rebars and prescribed loading and boundary conditions for continuum elements.

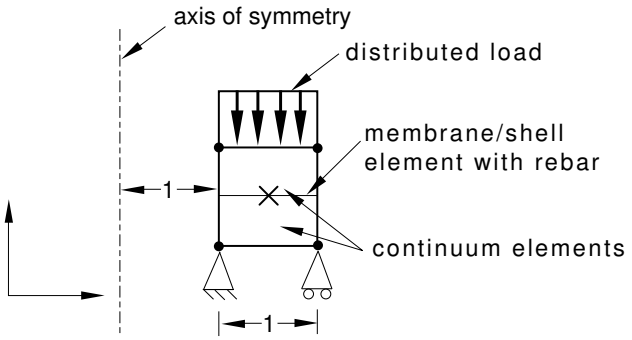


Figure 2: Axisymmetric model with rebars and prescribed loading and boundary conditions for shell and membrane elements.

A displacement of 0.1 is prescribed to the rigid body reference node of the continuum elements along the negative axial direction. A 360° model is generated with symmetric model generation. The axisymmetric results are read in as initial conditions using symmetric results transfer. Isoparametric, skew, and single rebars are verified with most elements. Rebar in embedded surface elements are also tested. Triangular and wedge elements are verified without rebars.

Results and discussion

The three-dimensional model generated is verified to be correct. The results from the axisymmetric analysis are transferred correctly onto the three-dimensional model.

Input files

pca3sfrev1.inp	CAX3 elements; two-dimensional model.
pc36sfrev1.inp	CAX3 elements; three-dimensional model.

pca3shrev1.inp	CAX3H elements; two-dimensional model.
pc36shrev1.inp	CAX3H elements; three-dimensional model.
pca4sfrev1.inp	CAX4 with SFMAX1 elements; two-dimensional model.
pc38sfrev1.inp	CAX4 with SFM3D4R elements; three-dimensional model.
pca4shrev1.inp	CAX4H with SFMAX1 elements; two-dimensional model.
pc38shrev1.inp	CAX4H with SFM3D4R elements; three-dimensional model.
pca4sirev1.inp	CAX4I with SFMAX1 elements; two-dimensional model.
pc38sirev1.inp	CAX4I with SFM3D4R elements; three-dimensional model.
pca4sjrev1.inp	CAX4IH with SFMAX1 elements; two-dimensional model.
pc38sjrev1.inp	CAX4IH with SFM3D4R elements; three-dimensional model.
pca4srrev1.inp	CAX4R with SFMAX1 elements; two-dimensional model.
pc38srrev1.inp	CAX4R with SFM3D4R elements; three-dimensional model.
pca4syrev1.inp	CAX4RH with SFMAX1 elements; two-dimensional model.
pc38syrev1.inp	CAX4RH with SFM3D4R elements; three-dimensional model.
pca6sfrev1.inp	CAX6 elements; two-dimensional model.
pc3fsfrev1.inp	CAX6 elements; three-dimensional model.
pca6shrev1.inp	CAX6H elements; two-dimensional model.
pc3fshrev1.inp	CAX6H elements; three-dimensional model.
pca8sfrev1.inp	CAX8 with SFMAX2 elements; two-dimensional model.
pc3ksfrev1.inp	CAX8 with SFM3D8R elements; three-dimensional model.
pca8shrev1.inp	CAX8H with SFMAX2 elements; two-dimensional model.
pc3kshrev1.inp	CAX8H with SFM3D8R elements; three-dimensional model.
pca8srrev1.inp	CAX8R with SFMAX2 elements; two-dimensional model.
pc3ksrrev1.inp	CAX8R with SFM3D8R elements; three-dimensional model.
pca3gfrev2.inp	CGAX3 elements; two-dimensional model.
pc36sfrev2.inp	CGAX3 elements; three-dimensional model.
pca3ghrev2.inp	CGAX3H elements; two-dimensional model.
pc36shrev2.inp	CGAX3H elements; three-dimensional model.
pca4gfrev2.inp	CGAX4 with SFMGAX1 elements; two-dimensional model.
pc38sfrev2.inp	CGAX4 with SFM3D4R elements; three-dimensional model.
pca4ghrev2.inp	CGAX4H with SFMGAX1 elements; two-dimensional model.
pc38shrev2.inp	CGAX4H with SFM3D4R elements; three-dimensional model.
pca4grrev2.inp	CGAX4R with SFMGAX1 elements; two-dimensional model.
pc38srrev2.inp	CGAX4R with SFM3D4R elements; three-dimensional model.
pca4gyrev2.inp	CGAX4RH with SFMGAX1 elements; two-dimensional model.
pc38syrev2.inp	CGAX4RH with SFM3D4R elements; three-dimensional model.
pca6gfrev2.inp	CGAX6 elements; two-dimensional model.

pc3fsfrev2.inp	CGAX6 elements; three-dimensional model.
pca6ghrev2.inp	CGAX6H elements; two-dimensional model.
pc3fshrev2.inp	CGAX6H elements; three-dimensional model.
pca8gfrev2.inp	CGAX8 with SFMGAX2 elements; two-dimensional model.
pc3ksfrev2.inp	CGAX8 with SFM3D8R elements; three-dimensional model.
pca8ghrev2.inp	CGAX8H with SFMGAX2 elements; two-dimensional model.
pc3kshrev2.inp	CGAX8H with SFM3D8R elements; three-dimensional model.
pca8gyrev2.inp	CGAX8RH with SFMGAX2 elements; two-dimensional model.
pc3ksyrev2.inp	CGAX8RH with SFM3D8R elements; three-dimensional model.
pma2srrev1.inp	MAX1 elements; two-dimensional model.
pm34srrev1.inp	MAX1 elements; three-dimensional model.
pma3srrev1.inp	MAX2 elements; two-dimensional model.
pm38srrev1.inp	MAX2 elements; three-dimensional model.
pmg2srrev2.inp	MGAX1 elements; two-dimensional model.
pm34srrev2.inp	MGAX1 elements; three-dimensional model.
pmg3srrev2.inp	MGAX2 elements; two-dimensional model.
pm38srrev2.inp	MGAX2 elements; three-dimensional model.
psa2sxrev1.inp	SAX1 elements; two-dimensional model.
psf4sxrev1.inp	SAX1 elements; three-dimensional model.
psa3sxrev1.inp	SAX2 elements; two-dimensional model.
ps68sxrev1.inp	SAX2 elements; three-dimensional model.
psa2dxrev1.inp	DSAX1 elements; two-dimensional model.
psf4dxrev1.inp	DSAX1 elements; three-dimensional model.
psa3dxrev1.inp	DSAX2 elements; two-dimensional model.
ps68dxrev1.inp	DSAX2 elements; three-dimensional model.
pca4tfrev1.inp	CAX4T with SFMAX1 elements; two-dimensional model.
pc38tfrev1.inp	CAX4T with SFM3D4R elements; three-dimensional model.
pca4threv1.inp	CAX4HT with SFMAX1 elements; two-dimensional model.
pc38threv1.inp	CAX4HT with SFM3D4R elements; three-dimensional model.
pca4trrev1.inp	CAX4RT with SFMAX1 elements; two-dimensional model.
pc38trrev1.inp	CAX4RT with SFM3D4R elements; three-dimensional model.
pca8tfrev1.inp	CAX8T with SFMAX2 elements; two-dimensional model.
pc3ktfrev1.inp	CAX8T with SFM3D8R elements; three-dimensional model.
pca8threv1.inp	CAX8HT with SFMAX2 elements; two-dimensional model.
pc3kthrev1.inp	CAX8HT with SFM3D8R elements; three-dimensional model.
pca8trrev1.inp	CAX8RT with SFMAX2 elements; two-dimensional model.
pc3ktrrev1.inp	CAX8RT with SFM3D8R elements; three-dimensional model.

pca8tyrev1.inp	CAX8RHT with SFMAX2 elements; two-dimensional model.
pc3ktyrev1.inp	CAX8RHT with SFM3D8R elements; three-dimensional model.
pca4dfrev1.inp	DCAX4 elements; two-dimensional model.
pc38dfrev1.inp	DCAX4 elements; three-dimensional model.
pca4tfrev2.inp	CGAX4T with SFMGAX1 elements; two-dimensional model.
pc38tfrev2.inp	CGAX4T with SFM3D4R elements; three-dimensional model.
pca8tfrev2.inp	CGAX8T with SFMGAX2 elements; two-dimensional model.
pc3ktfrev2.inp	CGAX8T with SFM3D8R elements; three-dimensional model.

Symmetric model generation by reflecting a partial three-dimensional model through a symmetry line

Elements tested

C3D6 C3D8I C3D8R C3D15 C3D20 C3D20R S4R DC3D8 C3D8RT

Features tested

Symmetric model generation by reflecting a partial three-dimensional model through a symmetry line and symmetric results transfer.

Problem description

Material:

Young's modulus	5×10^3
Poisson's ratio	0.3
Young's modulus (rebars)	2.5×10^5
Poisson's ratio (rebars)	0.0

Loading and boundary conditions: The loading and boundary conditions on the symmetric three-dimensional model are depicted in *Figure 3*. Internal pressure of 10 units is applied to the cylindrical model, while the top and bottom edges of the cylinder are clamped. The complete three-dimensional model is generated by reflecting the symmetric three-dimensional model about the axis shown. The symmetric results are read into the complete three-dimensional model as initial conditions. Wedge elements are verified without rebars.

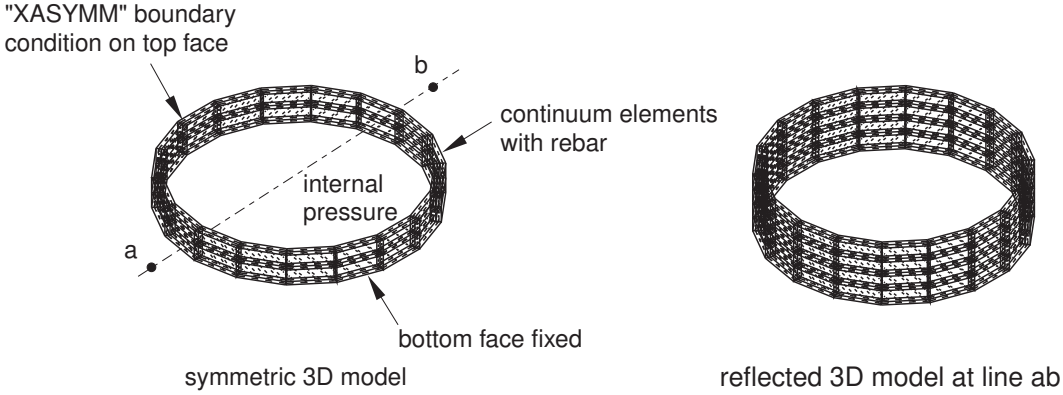


Figure 3: Symmetric 3D model with rebars and prescribed loading and boundary conditions.

Results and discussion

The reflected three-dimensional model generated is verified to be correct. The results from the symmetric three-dimensional analysis are transferred correctly onto the reflected three-dimensional model.

Input files

pca3sflin0.inp	C3D6 elements; two-dimensional model.
pc36sflin1.inp	C3D6 elements; symmetric three-dimensional model, perturbation step with **LOAD CASE*.
pc36sflin2.inp	C3D6 elements; reflected three-dimensional model, perturbation step with **LOAD CASE*.
pca4silin0.inp	C3D8I elements; two-dimensional model.
pc38silin1.inp	C3D8I elements; symmetric three-dimensional model.
pc38silin2.inp	C3D8I elements; reflected three-dimensional model.
pca4srlin0.inp	C3D8R elements; two-dimensional model.
pc38srlin1.inp	C3D8R elements; symmetric three-dimensional model.
pc38srlin2.inp	C3D8R elements; reflected three-dimensional model.
pca6sflin0.inp	C3D15 elements; two-dimensional model.
pc3fsflin1.inp	C3D15 elements; symmetric three-dimensional model.
pc3fsflin2.inp	C3D15 elements; reflected three-dimensional model.
pca8sflin0.inp	C3D20 elements; two-dimensional model.
pc3ksflin1.inp	C3D20 elements; symmetric three-dimensional model.
pc3ksflin2.inp	C3D20 elements; reflected three-dimensional model.
pca8srlin0.inp	C3D20R elements; two-dimensional model.
pc3ksrlin1.inp	C3D20R elements; symmetric three-dimensional model.
pc3ksrlin2.inp	C3D20R elements; reflected three-dimensional model.
psa2sslin0.inp	S4R elements; two-dimensional model.
psf4sslin1.inp	S4R elements; symmetric three-dimensional model.
psf4sslin2.inp	S4R elements; reflected three-dimensional model.
pca4trlin0.inp	C3D8RT elements; two-dimensional model.
pc38trlin1.inp	C3D8RT elements; symmetric three-dimensional model.
pc38trlin2.inp	C3D8RT elements; reflected three-dimensional model.
pca4dflin0.inp	DC3D8 elements; two-dimensional model.
pc38dflin1.inp	DC3D8 elements; symmetric three-dimensional model.
pc38dflin2.inp	DC3D8 elements; reflected three-dimensional model.

Symmetric model generation by reflecting a partial three-dimensional model through a symmetry plane

Elements tested

Continuum elements

C3D6H C3D8H C3D8RH C3D15H C3D20H C3D20RH DC3D8 C3D8HT

Shell and membrane elements

M3D3 M3D4 M3D4R M3D6 M3D8 M3D8R M3D9 M3D9R S3 S3R S4 S4R S4R5 S8R S8R5 STRI3 STRI65

Surface elements embedded in continuum elements

SFM3D4R SFM3D8R

Features tested

Symmetric model generation by reflecting a partial three-dimensional model through a symmetry plane and symmetric results transfer.

Problem description

Material properties for continuum elements:

Young's modulus	5×10^3
Poisson's ratio	0.4
Young's modulus (rebars)	1×10^6
Poisson's ratio (rebars)	0.0

Material properties for shell and membrane elements:

Young's modulus	1×10^4
Poisson's ratio	0.3
Young's modulus (rebars)	2.5×10^5
Poisson's ratio (rebars)	0.0

Loading and boundary conditions for continuum element model: The loading and boundary conditions on the symmetric three-dimensional model are depicted in *Figure 4*.

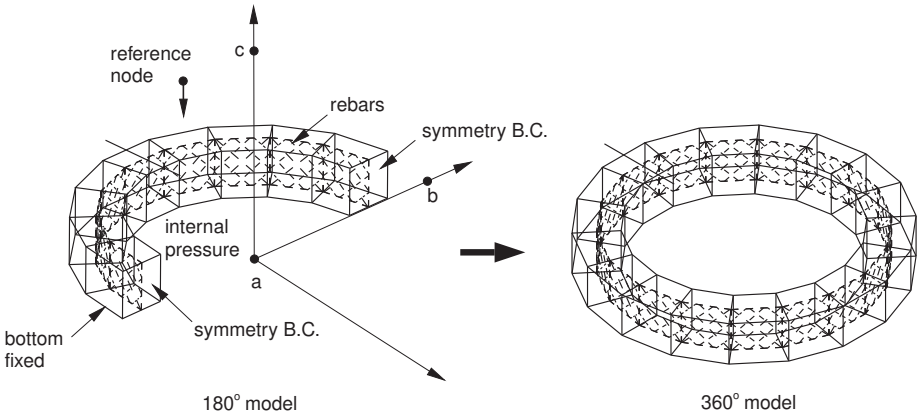


Figure 4: Symmetric and reflected 3D continuum element model with prescribed boundary conditions.

The rigid surface reference node is displaced by 0.05 units along the negative axial direction. The complete three-dimensional model is generated by reflecting the symmetric three-dimensional model about the x–z plane.

The symmetric results are read into the complete three-dimensional model as initial conditions. Wedge elements are verified without rebars.

Loading and boundary conditions for shell and membrane element models: The loading and boundary conditions on the symmetric model are depicted in *Figure 5*. The complete three-dimensional model is generated by reflecting the symmetric three-dimensional model about the y–z plane. The symmetric results are read into the complete three-dimensional model as initial conditions. Elements S3/S3R, S4R5, S8R5, STRI3, and STRI65 are verified. Rebars are not defined in triangular elements.

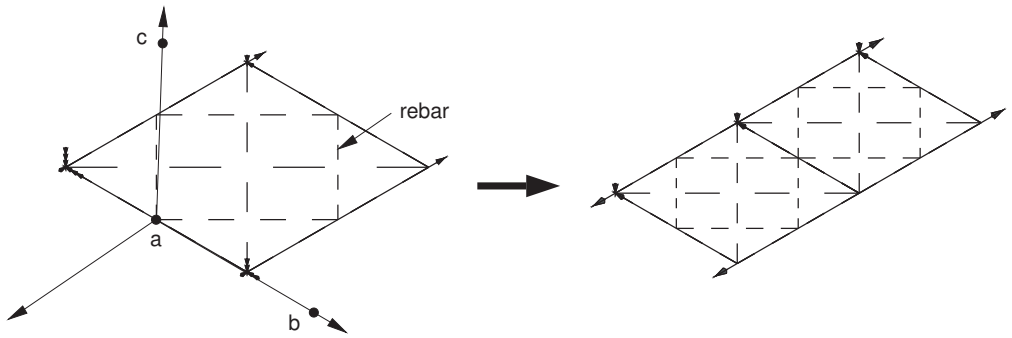


Figure 5: Symmetric shell and membrane model with rebars and prescribed boundary conditions.

Results and discussion

The reflected three-dimensional model generated is verified to be correct. The results from the symmetric three-dimensional analysis are transferred correctly onto the reflected three-dimensional model.

Input files

pca3shpln0.inp	C3D6H elements; two-dimensional model.
pca3shpln0_surf.inp	C3D6H elements; two-dimensional model using surface-to-surface contact.
pc36shpln1.inp	C3D6H elements; symmetric three-dimensional model.
pc36shpln2.inp	C3D6H elements; reflected three-dimensional model.
pca4shpln0.inp	C3D8H with SFMAX1 elements; two-dimensional model.
pc38shpln1.inp	C3D8H with SFM3D4R elements; symmetric three-dimensional model.
pc38shpln2.inp	C3D8H with SFM3D4R elements; reflected three-dimensional model.
pca4sypln0.inp	C3D8RH with SFMAX1 elements; two-dimensional model.
pc38sypln1.inp	C3D8RH with SFM3D4R elements; symmetric three-dimensional model.
pc38sypln2.inp	C3D8RH with SFM3D4R elements; reflected three-dimensional model.
pca6shpln0.inp	C3D15H elements; two-dimensional model.
pc3fshpln1.inp	C3D15H elements; symmetric three-dimensional model.
pc3fshpln2.inp	C3D15H elements; reflected three-dimensional model.
pca8shpln0.inp	C3D20H with SFMAX2 elements; two-dimensional model.
pc3kshpln1.inp	C3D20H with SFM3D8R elements; symmetric three-dimensional model.
pc3kshpln2.inp	C3D20H with SFM3D8R elements; reflected three-dimensional model.

pca8sypln0.inp	C3D20RH with SFMAX2 elements; two-dimensional model.
pc3ksypln1.inp	C3D20RH with SFM3D8R elements; symmetric three-dimensional model.
pc3ksypln2.inp	C3D20RH with SFM3D8R elements; reflected three-dimensional model.
pm33sfpln1.inp	M3D3 elements; symmetric three-dimensional model.
pm33sfpln2.inp	M3D3 elements; reflected three-dimensional model.
pm34sfpln1.inp	M3D4 elements; symmetric three-dimensional model.
pm34sfpln2.inp	M3D4 elements; reflected three-dimensional model.
pm34srpln1.inp	M3D4R elements; symmetric three-dimensional model.
pm34srpln2.inp	M3D4R elements; reflected three-dimensional model.
pm36sfpln1.inp	M3D6 elements; symmetric three-dimensional model.
pm36sfpln2.inp	M3D6 elements; reflected three-dimensional model.
pm3dsfpln1.inp	M3D8 elements; symmetric three-dimensional model.
pm3dsfpln2.inp	M3D8 elements; reflected three-dimensional model.
pm3dsrpln1.inp	M3D8R elements; symmetric three-dimensional model.
pm3dsrpln2.inp	M3D8R elements; reflected three-dimensional model.
pm39sfpln1.inp	M3D9 elements; symmetric three-dimensional model.
pm39sfpln2.inp	M3D9 elements; reflected three-dimensional model.
pm39srpln1.inp	M3D9R elements; symmetric three-dimensional model.
pm39srpln2.inp	M3D9R elements; reflected three-dimensional model.
psf3sspln1.inp	S3/S3R elements; symmetric three-dimensional model.
psf3sspln2.inp	S3/S3R elements; reflected three-dimensional model.
pse4sspln1.inp	S4 elements; symmetric three-dimensional model.
pse4sspln2.inp	S4 elements; reflected three-dimensional model.
psf4sspln1.inp	S4R elements; symmetric three-dimensional model.
psf4sspln2.inp	S4R elements; reflected three-dimensional model.
ps54sspln1.inp	S4R5 elements; symmetric three-dimensional model.
ps54sspln2.inp	S4R5 elements; reflected three-dimensional model.
ps68sspln1.inp	S8R elements; symmetric three-dimensional model.
ps68sspln2.inp	S8R elements; reflected three-dimensional model.
ps58sspln1.inp	S8R5 elements; symmetric three-dimensional model.
ps58sspln2.inp	S8R5 elements; reflected three-dimensional model.
ps63sspln1.inp	STRI3 elements; symmetric three-dimensional model.
ps63sspln2.inp	STRI3 elements; reflected three-dimensional model.
ps56sspln1.inp	STRI65 elements; symmetric three-dimensional model.
ps56sspln2.inp	STRI65 elements; reflected three-dimensional model.
pca4thpln0.inp	C3D8HT elements; two-dimensional model.
pc38thpln1.inp	C3D8HT elements; symmetric three-dimensional model.

pc38thpln2.inp	C3D8HT elements; reflected three-dimensional model.
pca4dfpln0.inp	DC3D8 elements; two-dimensional model.
pc38dfpln1.inp	DC3D8 elements; symmetric three-dimensional model.
pc38dfpln2.inp	DC3D8 elements; reflected three-dimensional model.

Symmetric model generation by revolving a single three-dimensional sector of the model about the symmetry axis

Elements tested

Continuum elements

C3D8 C3D20 C3D8T DC3D8

Shell and membrane elements

S4 M3D4R DS4

Features tested

Symmetric model generation by revolving a single three-dimensional sector of the model about the symmetry axis and symmetric results transfer.

Problem description

These tests verify the symmetric model generation and results transfer capability for a periodic structure. A three-dimensional periodic model is generated by revolving a three-dimensional repetitive sector about a symmetry axis. The bottom surface of the periodic model is fixed, while the top surface of the periodic model is in contact with a pad that is subjected to distributed loadings. If the symmetric surfaces in the original sector have precisely matched meshes, duplicated nodes will be eliminated automatically to ensure that the mesh is connected properly between the neighboring sectors when the original sector is revolved about the symmetry axis to create a periodic model. In all other cases constraints between the automatically generated neighboring pairs of corresponding surfaces are then applied with the automatically generated tie constraint when the original sector is revolved about the symmetry axis to create a periodic model. Both open (the structure has end edges) and closed loop periodic structures are considered. The results from the original sector are transferred to the periodic model.

Material properties:

Young's modulus	7×10^4
Poisson's ratio	0.33

Results and discussion

The three-dimensional periodic model generated is verified to be correct, as are the constraints between the neighboring pairs of corresponding surfaces when meshes for the symmetric surfaces are not matched precisely in the original sector. The results from the original sector are transferred correctly onto the periodic three-dimensional model.

Input files

smg_wedge.inp	C3D8 elements; a single three-dimensional sector with completely matched meshes.
smg_wedge_surf.inp	C3D8 elements; a single three-dimensional sector with completely matched meshes and surface-to-surface contact.

smg_period_open.inp	C3D8 elements; periodic three-dimensional model with open end edges; requires smg_wedge.inp.
smg_period_close.inp	C3D8 elements; periodic three-dimensional model with closed loop; requires smg_wedge.inp.
smg_noperiod_open.inp	C3D8 elements; variable sector angle in periodic three-dimensional model with open end edges; requires smg_wedge.inp.
smg_noperiod_close.inp	C3D8 elements; variable sector angle in periodic three-dimensional model with closed loop; requires smg_wedge.inp.
smg_period_open_surf.inp	C3D8 elements; periodic three-dimensional model with open end edges using surface-to-surface–based contact; requires smg_wedge_surf.inp.
smg_period_close_surf.inp	C3D8 elements; periodic three-dimensional model with closed loop using surface-to-surface–based contact; requires smg_wedge_surf.inp.
smg_wedge2.inp	C3D8 with S4 elements; a single three-dimensional sector with completely matched meshes.
smg_period_open2.inp	C3D8 with S4 elements; periodic three-dimensional model with open end edges; requires smg_wedge2.inp.
smg_period_close2.inp	C3D8 with S4 elements; periodic three-dimensional model with closed loop; requires smg_wedge2.inp.
smg_wedge3.inp	C3D8 with M3D4R elements; a single three-dimensional sector with completely matched meshes.
smg_period_open3.inp	C3D8 with M3D4R elements; periodic three-dimensional model with open end edges; requires smg_wedge3.inp.
smg_period_close3.inp	C3D8 with M3D4R elements; periodic three-dimensional model with closed loop; requires smg_wedge3.inp.
smg_wedge4.inp	C3D20 elements; a single three-dimensional sector with completely matched meshes.
smg_period_open4.inp	C3D20 elements; periodic three-dimensional model with open end edges; requires smg_wedge4.inp.
smg_period_close4.inp	C3D20 elements; periodic three-dimensional model with closed loop; requires smg_wedge4.inp.
smg_noperiod_open4.inp	C3D20 elements; variable sector angle in periodic three-dimensional model with open end edges; requires smg_wedge4.inp.
smg_noperiod_close4.inp	C3D20 elements; variable sector angle in periodic three-dimensional model with closed loop; requires smg_wedge4.inp.
smg_wedge4_surf.inp	C3D20 elements; a single three-dimensional sector with completely matched meshes using surface-to-surface–based contact.
smg_period_open4_surf.inp	C3D20 elements; periodic three-dimensional model with open end edges using surface-to-surface–based contact; requires smg_wedge4_surf.inp.
smg_period_close4_surf.inp	C3D20 elements; periodic three-dimensional model with closed loop using surface-to-surface–based contact; requires smg_wedge4_surf.inp.
smg_wedge_mismap.inp	C3D8 elements; a single three-dimensional sector with mismatched meshes.
smg_wedge_mismap_surf.inp	C3D8 elements; a single three-dimensional sector with mismatched meshes and surface-to-surface contact.
smg_period_open_mismap1.inp	C3D8 elements; periodic three-dimensional model with mismatched meshes and open end edges; requires smg_wedge_mismap.inp.
smg_period_open_mismap2.inp	C3D8 elements; periodic three-dimensional model with mismatched meshes and open end edges; requires smg_wedge_mismap.inp.

smg_period_close_mismap1.inp	C3D8 elements; periodic three-dimensional model with mismatched meshes and closed loop; requires smg_wedge_mismap.inp.
smg_period_close_mismap2.inp	C3D8 elements; periodic three-dimensional model with mismatched meshes and closed loop; requires smg_wedge_mismap.inp.
smg_noperiod_open_mismap1.inp	C3D8 elements; variable sector angle in periodic three-dimensional model with mismatched meshes and open end edges; requires smg_wedge_mismap.inp.
smg_noperiod_open_mismap2.inp	C3D8 elements; variable sector angle in periodic three-dimensional model with mismatched meshes and open end edges; requires smg_wedge_mismap.inp.
smg_noperiod_close_mismap1.inp	C3D8 elements; variable sector angle in periodic three-dimensional model with mismatched meshes and closed loop; requires smg_wedge_mismap.inp.
smg_noperiod_close_mismap2.inp	C3D8 elements; variable sector angle in periodic three-dimensional model with mismatched meshes and closed loop; requires smg_wedge_mismap.inp.
smg_wedge_mismap_surf.inp	C3D8 elements; a single three-dimensional sector with mismatched meshes using surface-to-surface–based contact.
smg_period_open_mismap1_surf.inp	C3D8 elements; periodic three-dimensional model with mismatched meshes and open end edges using surface-to-surface–based contact; requires smg_wedge_mismap_surf.inp.
smg_period_open_mismap2_surf.inp	C3D8 elements; periodic three-dimensional model with mismatched meshes and open end edges using surface-to-surface–based contact; requires smg_wedge_mismap_surf.inp.
smg_period_close_mismap1_surf.inp	C3D8 elements; periodic three-dimensional model with mismatched meshes and closed loop using surface-to-surface–based contact; requires smg_wedge_mismap_surf.inp.
smg_period_close_mismap2_surf.inp	C3D8 elements; periodic three-dimensional model with mismatched meshes and closed loop using surface-to-surface–based contact; requires smg_wedge_mismap_surf.inp.
smg_wedge-d1.inp	DC3D8 with DS4 elements; a single three-dimensional sector with completely matched meshes.
smg_period_open-d1.inp	DC3D8 with DS4 elements; periodic three-dimensional model with open end edges; requires smg_wedge-d1.inp.
smg_wedge_mismap-d1.inp	DC3D8 elements; a single three-dimensional sector with mismatched meshes.
smg_period_open_mismap-d1.inp	DC3D8 elements; periodic three-dimensional model with mismatched meshes and open end edges; requires smg_wedge_mismap-d1.inp.
smg_wedge_mismap-t1.inp	C3D8T elements; a single three-dimensional sector with mismatched meshes.
smg_period_open_mismap-t1.inp	C3D8T elements; periodic three-dimensional model with mismatched meshes and open end edges; requires smg_wedge_mismap-t1.inp.

Symmetric model generation and results transfer with large deformation

Elements tested

Continuum elements

CGAX4 CGAX4H CGAX4RH

Membrane and surface elements
MGAX1 SFMGAX1

Features tested

Symmetric model generation and results transfer for models involving large deformation, frictional contact with a curved surface, rebars, embedded elements, and surface elements with rebar layers.

Problem description

These tests verify the symmetric model generation and results transfer capability for a hyperelastic rubberlike material reinforced by stiff strands. The strands are modeled either as rebars directly in continuum elements, as rebar layers in membrane elements embedded in continuum elements, or as rebar layers in surface elements embedded in continuum elements. The model consists of a Mooney-Rivlin material, and the reinforcing strands are linear elastic. The strands have a cross-sectional area of 0.5 square mm each, are laid in a single layer with a spacing of 5 mm, and are inclined at 50° to the r–z plane in the axisymmetric model. The reinforced body is then compressed along the z-direction by rigid curved surfaces resulting in large deformations in the material. The strands do not lie in the r–z plane; therefore, this compression results in twisting of the material about the axis of symmetry.

A three-dimensional revolved model is generated from the axisymmetric model, and the results from the axisymmetric analysis are transferred to the revolved model. The three-dimensional revolved model is then reflected through a line, and the results are transferred to this reflected model.

Material:

C10	2.0×10^6 Pa
C01	1.5×10^6 Pa
D1 (incompressible)	0.0
D1 (compressible)	1.452×10^{-8} Pa^{-1}
Coefficient of friction	0.1
Young's modulus(rebars)	2.0×10^{11} Pa

Results and discussion

The undeformed and deformed axisymmetric model is depicted in *Figure 6*. A displacement of 0.08 has been prescribed to the rigid body reference node along the negative axial direction. A cut-out of the deformed three-dimensional revolved and reflected model is shown in *Figure 7*. The three-dimensional model and the transferred results are verified to be correct.

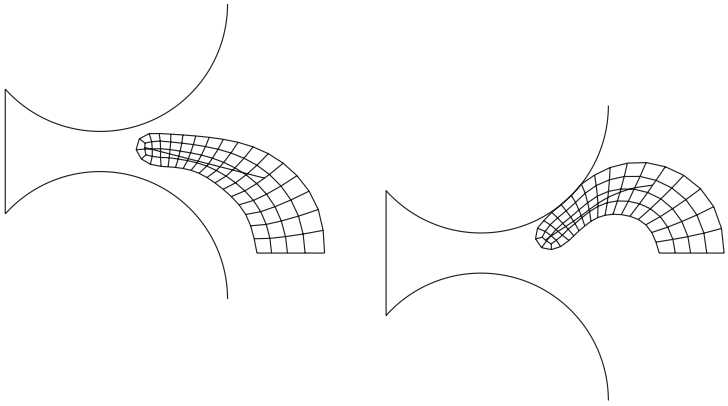


Figure 6: Undeformed and deformed axisymmetric model with rebar layer.

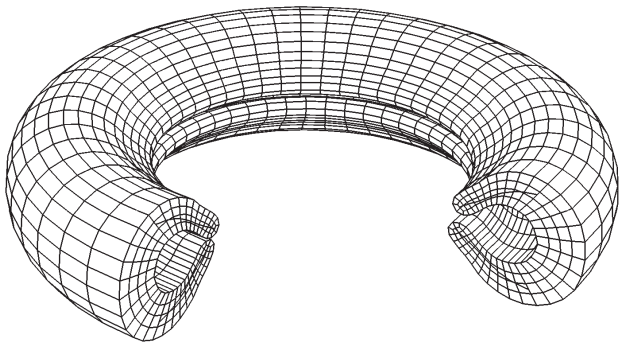


Figure 7: Cut-out of the revolved and reflected three-dimensional model.

Input files

pca4gfreb0.inp	CGAX4 elements; axisymmetric model with rebars.
pc38sfreb1.inp	C3D8 elements; three-dimensional revolved model.
pc38sfreb2.inp	C3D8 elements; three-dimensional reflected model.
pca4ghreb0.inp	CGAX4H elements; axisymmetric model with rebars.
pc38shreb1.inp	C3D8H elements; three-dimensional revolved model.
pc38shreb2.inp	C3D8H elements; three-dimensional reflected model.
pca4gyreb0.inp	CGAX4RH elements; axisymmetric model with rebars.
pc38syreb1.inp	C3D8RH elements; three-dimensional revolved model.
pc38syreb2.inp	C3D8RH elements; three-dimensional reflected model.
pca4gfmem0.inp	CGAX4 elements; axisymmetric model with rebar layers in embedded membranes.
pc38sfmem1.inp	C3D8 elements; three-dimensional revolved model.
pc38sfmem2.inp	C3D8 elements; three-dimensional reflected model.
pc38sfmem5.inp	CCL12 elements; three-dimensional revolved model.
pc38sfmem6.inp	CCL12 elements; three-dimensional reflected model.
pca4ghmem0.inp	CGAX4H elements; axisymmetric model with rebar layers in embedded membranes.

pc38shmem1.inp	C3D8H elements; three-dimensional revolved model.
pc38shmem2.inp	C3D8H elements; three-dimensional reflected model.
pc38shmem5.inp	CCL12H elements; three-dimensional revolved model.
pc38shmem6.inp	CCL12H elements; three-dimensional reflected model.
pca4gymem0.inp	CGAX4RH elements; axisymmetric model with rebar layers in embedded membranes.
pc38symem1.inp	C3D8RH elements; three-dimensional revolved model.
pc38symem2.inp	C3D8RH elements; three-dimensional reflected model.
pca4gfsrf0.inp	CGAX4 elements; axisymmetric model with rebar layers in embedded surface elements.
pc38sfsrf1.inp	C3D8 elements; three-dimensional revolved model.
pc38sfsrf2.inp	C3D8 elements; three-dimensional reflected model.
pc38sfsrf5.inp	CCL12 elements; three-dimensional revolved model.
pc38sfsrf6.inp	CCL12 elements; three-dimensional reflected model.
pca4ghsrf0.inp	CGAX4H elements; axisymmetric model with rebar layers in embedded surface elements.
pc38shsrf1.inp	C3D8H elements; three-dimensional revolved model.
pc38shsrf2.inp	C3D8H elements; three-dimensional reflected model.
pc38shsrf5.inp	CCL12H elements; three-dimensional revolved model.
pc38shsrf6.inp	CCL12H elements; three-dimensional reflected model.
pca4gysrf0.inp	CGAX4RH elements; axisymmetric model with rebar layers in embedded surface elements.
pc38sysrf1.inp	C3D8RH elements; three-dimensional revolved model.
pc38sysrf2.inp	C3D8RH elements; three-dimensional reflected model.

Analysis of cyclic symmetric models

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

B21 B22 B32
C3D8R C3D20 CPE4H CPE4R CPE8R CPS4R CPS8R
DC2D4 DC2D8 DC3D8
S4R S8R SC8R CSS8

Features tested

Natural frequency extraction for two-dimensional and three-dimensional models that exhibit cyclic symmetry.

Element-based and node-based cyclic symmetric surface definitions on matched and mismatched meshes.

Nodal transformation with the cyclic symmetry analysis capability and use of multi-point constraints.

Application of preload prior to natural frequency extraction.

Modal-based steady-state dynamic analysis for models that exhibit cyclic symmetry.

Heat transfer analysis for models that exhibit cyclic symmetry.

Problem description

The models consist of 1×2 and 2×2 element meshes. There are no boundary conditions and loads, except for preload tests. Preloading of the model is done with both concentrated and distributed loads and for heat transfer analysis with temperature assigned to the chosen set of nodes.

Results and discussion

The results for the natural frequency extraction of the cyclic symmetric models are the same as those obtained for a corresponding 360° model.

Input files

Beam elements

cyclicsym_b21_nn.inp	Two-element mesh with node-type secondary surface and node-type main surface.
cyclicsym_b21_360.inp	Full model corresponding to two-element mesh of B21 elements.
cyclicsym_b22_nn.inp	Single-element mesh with node-type secondary surface and node-type main surface.
cyclicsym_b22_360.inp	Full model corresponding to single-element mesh of B22 elements.
cyclicsym_b31_nn.inp	Single-element mesh with node-type secondary surface and node-type main surface.

cyclicsym_b31_360.inp	Full model corresponding to single-element mesh of B31 elements.
cyclicsym_b32_nn.inp	Single-element mesh with node-type secondary surface and node-type main surface.
cyclicsym_b32_360.inp	Full model corresponding to single-element mesh of B32 elements.

Continuum elements

cyclicsym_c3d8r_ee.inp	1 × 2 mesh with element-type secondary surface and element-type main surface.
cyclicsym_c3d8r_en.inp	1 × 2 mesh with element-type secondary surface and node-type main surface.
cyclicsym_c3d8r_ne.inp	1 × 2 mesh with node-type secondary surface and element-type main surface.
cyclicsym_c3d8r_nn.inp	1 × 2 mesh with node-type secondary surface and node-type main surface.
cyclicsym_c3d8r_360.inp	Full model corresponding to 1 × 2 mesh of C3D8R elements.
cyclicsym_c3d8r_nn_ref.inp	2 × 2 mesh with node-type secondary surface and node-type main surface.
cyclicsym_c3d8r_360_ref.inp	Full model corresponding to 2 × 2 mesh of C3D8R elements.
cyclicsym_c3d8r_mis_map_ee.inp	2 × 2 mismatched mesh with element-type secondary surface and element-type main surface.
cyclicsym_c3d8r_mis_map_ne.inp	2 × 2 mismatched mesh with node-type secondary surface and element-type main surface.
cyclicsym_c3d8r_mis_map_360.inp	Full model corresponding to 2 × 2 mismatched mesh of C3D8R elements.
cyclicsym_c3d20_ne.inp	1 × 2 mesh with node-type secondary surface and element-type main surface.
cyclicsym_c3d20_nn.inp	1 × 2 mesh with node-type secondary surface and node-type main surface.
cyclicsym_c3d20_360.inp	Full model corresponding to 1 × 2 mesh of C3D20 elements.
cyclicsym_c3d20_nn_ref.inp	2 × 2 mesh with node-type secondary surface and node-type main surface.
cyclicsym_c3d20_360_ref.inp	Full model corresponding to 2 × 2 mesh of C3D20 elements.
cyclicsym_cpe4h_nn.inp	1 × 2 mesh with node-type secondary surface and node-type main surface.
cyclicsym_cpe4h_360.inp	Full model corresponding to 1 × 2 mesh of CPE4H elements.
cyclicsym_cpe4r_ee.inp	1 × 2 mesh with element-type secondary surface and element-type main surface.
cyclicsym_cpe4r_en.inp	1 × 2 mesh with element-type secondary surface and node-type main surface.
cyclicsym_cpe4r_ne.inp	1 × 2 mesh with node-type secondary surface and element-type main surface type.
cyclicsym_cpe4r_nn.inp	1 × 2 mesh with node-type secondary surface and node-type main surface.
cyclicsym_cpe4r_360.inp	Full model corresponding to 1 × 2 mesh of CPE4R elements.
cyclicsym_cpe4r_nn_ref.inp	2 × 2 mesh with node-type secondary surface and node-type main surface.

cyclicsym_cpe4r_360_ref.inp	Full model corresponding to 2 × 2 mesh of CPE4R elements.
cyclicsym_cpe4r_mis_map_ee.inp	2 × 2 mismatched mesh with element-type secondary surface and element-type main surface.
cyclicsym_cpe4r_mis_map_ne.inp	2 × 2 mismatched mesh with node-type secondary surface and element-type main surface type.
cyclicsym_cpe4r_mis_map_360.inp	Full model corresponding to 2 × 2 mismatched mesh of CPE4R elements.
cyclicsym_cpe8r_ee.inp	1 × 2 mesh with element-type secondary surface and element-type main surface.
cyclicsym_cpe8r_en.inp	1 × 2 mesh with element-type secondary surface and node-type main surface.
cyclicsym_cpe8r_ne.inp	1 × 2 mesh with node-type secondary surface and element-type main surface.
cyclicsym_cpe8r_nn.inp	1 × 2 mesh with node-type secondary surface and node-type main surface.
cyclicsym_cpe8r_360.inp	Full model corresponding to 1 × 2 mesh of CPE8R elements.
cyclicsym_cpe8r_nn_ref.inp	2 × 2 mesh with node-type secondary surface and node-type main surface.
cyclicsym_cpe8r_360_ref.inp	Full model corresponding to 2 × 2 mesh of CPE8R elements.
cyclicsym_cps4r_ee.inp	1 × 2 mesh with element-type secondary surface and element-type main surface.
cyclicsym_cps4r_en.inp	1 × 2 mesh with element-type secondary surface and node-type main surface.
cyclicsym_cps4r_ne.inp	1 × 2 mesh with node-type secondary surface and element-type main surface.
cyclicsym_cps4r_nn.inp	1 × 2 mesh with node-type secondary surface and node-type main surface.
cyclicsym_cps4r_360.inp	Full model corresponding to 1 × 2 mesh of CPS4R elements.
cyclicsym_cps4r_nn_ref.inp	2 × 2 mesh with node-type secondary surface and node-type main surface.
cyclicsym_cps4r_360_ref.inp	Full model corresponding to 2 × 2 mesh of CPS4R elements.
cyclicsym_cps4r_mis_map_ee.inp	2 × 2 mismatched mesh with element-type secondary surface and element-type main surface.
cyclicsym_cps4r_mis_map_ne.inp	2 × 2 mismatched mesh with node-type secondary surface and element-type main surface type.
cyclicsym_cps4r_mis_map_360.inp	Full model corresponding to 2 × 2 mismatched mesh of CPS4R elements.
cyclicsym_cps8r_ee.inp	1 × 2 mesh with element-type secondary surface and element-type main surface.
cyclicsym_cps8r_en.inp	1 × 2 mesh with element-type secondary surface and node-type main surface.
cyclicsym_cps8r_ne.inp	1 × 2 mesh with node-type secondary surface and element-type main surface.
cyclicsym_cps8r_nn.inp	1 × 2 mesh with node-type secondary surface and node-type main surface.
cyclicsym_cps8r_360.inp	Full model corresponding to 1 × 2 mesh of CPS8R elements.

cyclicsym_cps8r_nn_ref.inp	2 × 2 mesh with node-type secondary surface and node-type main surface.
cyclicsym_cps8r_360_ref.inp	Full model corresponding to 2 × 2 mesh of CPS8R elements.
cyclicsym_css8_nn.inp	1 × 2 mesh with node-type secondary surface and node-type main surface.
cyclicsym_css8_360.inp	Full model corresponding to 1 × 2 mesh of CSS8 elements.

Continuum shell elements

cyclicsym_sc8r_nn.inp	1 × 2 mesh with node-type secondary surface and node-type main surface.
cyclicsym_sc8r_360.inp	Full model corresponding to 1 × 2 mesh of SC8R elements.

Shell elements

cyclicsym_s4r_nn.inp	1 × 2 mesh with node-type secondary surface and node-type main surface.
cyclicsym_s4r_360.inp	Full model corresponding to 1 × 2 mesh of S4R elements.
cyclicsym_s4r_nn_ref.inp	2 × 2 mesh with node-type secondary surface and node-type main surface.
cyclicsym_s4r_360_ref.inp	Full model corresponding to 2 × 2 mesh of S4R elements.
cyclicsym_s4r_mis_map_ne.inp	2 × 2 mismatched mesh with node-type secondary surface and element-type main surface.
cyclicsym_s4r_mis_map_360.inp	Full model corresponding to 12 × 2 mismatched mesh of S4R elements.
cyclicsym_s8r_nn.inp	1 × 2 mesh with node-type secondary surface and node-type main surface.
cyclicsym_s8r_360.inp	Full model corresponding to 1 × 2 mesh of S8R elements.
cyclicsym_s8r_nn_ref.inp	2 × 2 mesh with node-type secondary surface and node-type main surface.
cyclicsym_s8r_360_ref.inp	Full model corresponding to 2 × 2 mesh of S8R elements.

Cyclic Symmetry in Explicit Dynamic Analysis

2d_cyclic_xpl.inp	Cyclic symmetry model with plane stress element.
3d_cyclic_xpl.inp	Cyclic symmetry model with 3D continuum elements.
shell_cyclic_xpl.inp	Cyclic symmetry model with shell S4R elements.
Tube_cyclic_xpl.inp	Cyclic symmetry model with acoustic elements tied to shell elements.

Other features tested

cyclicsym_c3d8r_ee_cload_x.inp	Frequency extraction after a preloading step in which a **CLOAD* is applied in the global *x*-direction.
cyclicsym_c3d8r_ee_cload_y.inp	Frequency extraction after a preloading step in which a **CLOAD* is applied in the global *y*-direction.
cyclicsym_c3d8r_360_cload.inp	Full model corresponding to the application of a **CLOAD* before frequency extraction.
cyclicsym_c3d8r_ee_dload.inp	Frequency extraction after a preloading step in which a **DLOAD* is applied.

cyclicsym_c3d8r_360_dload.inp	Full model corresponding to the application of a **DLOAD* before frequency extraction.
cyclicsym_c3d8r_nn_mpc.inp	Cyclic symmetric model with an **MPC* option.
cyclicsym_c3d8r_360_mpc.inp	Full model corresponding to the use of an **MPC* option.
cyclicsym_c3d8r_nn_single_node_x.inp	Cyclic symmetric model with a single node at one end and the cyclic symmetry axis parallel to the global *x*-axis.
cyclicsym_c3d8r_nn_single_node_y.inp	Cyclic symmetric model with a single node at one end and the cyclic symmetry axis parallel to the global *y*-axis.
cyclicsym_c3d8r_nn_single_node_z.inp	Cyclic symmetric model with a single node at one end and the cyclic symmetry axis parallel to the global *z*-axis.
cyclicsym_c3d8r_360_single_node.inp	Full model corresponding to the cyclic symmetric model with a single node at one end.
cyclicsym_c3d8r_nn_trans.inp	Cyclic symmetric model with a **TRANSFORM* option.
cyclicsym_c3d8r_360_trans.inp	Full model corresponding to the use of a **TRANSFORM* option.
cyclicsym_c3d8r_ss.inp	Cyclic symmetric model with a preloading step using the **VISCO* option, frequency extraction, and steady-state modal-based dynamic analysis.
cyclicsym_c3d8r_360_ss.inp	Full model corresponding to the use of a preloading step with the **VISCO* option, frequency extraction, and steady-state modal-based dynamic analysis.
cyclicsym_cpe4r_nn_basis.inp	Cyclic symmetric model with a static nonlinear step that forms the basis for the analysis in cyclicsym_cpe4r_nn_rezone.inp.
cyclicsym_cpe4r_nn_rezone.inp	Cyclic symmetric model with a frequency extraction step following the mapping of the solution from the analysis in cyclicsym_cpe4r_nn_basis.inp.
cyclicsym_cpe4r_360_basis.inp	Full model with a static nonlinear step that forms the basis for the analysis in cyclicsym_cpe4r_360_rezone.inp.
cyclicsym_cpe4r_360_rezone.inp	Full model with a frequency extraction step following the mapping of the solution from the analysis in cyclicsym_cpe4r_360_basis.inp.
cyclicsym_cpe4r_nn_multi_step.inp	Cyclic symmetric model with multiple frequency extraction steps.
cyclicsym_cpe4r_nn_restart.inp	Cyclic symmetric model with multiple frequency extraction steps performed as a restart analysis after the first step in the analysis in cyclicsym_cpe4r_nn_multi_step.inp.
cyclicsym_cps4r_nn_sub.inp	Cyclic symmetric model with the **SUBMODEL* option.
cyclicsym_cps4r_nn_sub_sb.inp	Cyclic symmetric model with the **SUBMODEL*, TYPE=SURFACE option.
cyclicsym_cps4r_nn_full.inp	Full cyclic symmetric model used in the analysis in cyclicsym_cps4r_nn_sub.inp and cyclicsym_cps4r_nn_sub_sb.inp.
cyclicsym_s4r_nn_mdl_change.inp	Cyclic symmetric model with the **MODEL CHANGE* option.
cyclicsym_s4r_360_mdl_change.inp	Full model with the **MODEL CHANGE* option.
cyclicsym_cpe4r_ss.inp	Cyclic symmetric model with a nonlinear preloading static step, frequency extraction, and modal-based steady-state dynamic analysis (one nodal diameter) using the **DLOAD* and **DSLOAD* options.
cyclicsym_cpe4r_360_ss.inp	Full model with a nonlinear preloading static step, frequency extraction, and modal-based steady-state dynamic analysis (one nodal diameter) using the **DLOAD* and **DSLOAD* options.

cyclicsym_cpe4r_ss2.inp	Cyclic symmetric model with frequency extraction and modal-based steady-state dynamic analysis (two nodal diameters) using the **DLOAD* and **DSLOAD* options.
cyclicsym_cpe4r_360_ss2.inp	Full model with frequency extraction and modal-based steady-state dynamic analysis (two nodal diameters) using the **DLOAD* and **DSLOAD* options.
cyclicsym_cpe4r_cl_ss.inp	Cyclic symmetric model with a nonlinear preloading static step, frequency extraction, and two modal-based steady-state dynamic analyses (one and two nodal diameters) using the **CLOAD*, **DLOAD*, and **TRANSFORM* options.
cyclicsym_cpe4r_360_cl_ss.inp	Full model with a nonlinear preloading static step, frequency extraction, and two modal-based steady-state dynamic analyses (one and two nodal diameters) using the **CLOAD*, **DLOAD*, and **TRANSFORM* options.
cyclicsym_dc2d4_mis_heat.inp	Cyclic symmetric model with a transient heat transfer analysis.
cyclicsym_dc2d4_mis_heat_360.inp	Full model with a transient heat transfer analysis.
cyclicsym_dc2d8_nn_heat.inp	Cyclic symmetric model with a steady-state heat transfer analysis.
cyclicsym_dc2d8_heat_360.inp	Full model with a steady-state heat transfer analysis.
cyclicsym_dc3d8_nn_heat.inp	Cyclic symmetric model with a steady-state heat transfer analysis.
cyclicsym_dc3d8_heat_360.inp	Full model with a steady-state heat transfer analysis.
cyclicsym_dcc2d4_nn_loads.inp	Cyclic symmetric model with all thermal loads (**FILM*, **CFILM*, **SFILM*, **CFLUX*, **DFLUX*, **DSFLUX*, **RADIATE*, **CRADIATE*, and **SRADIATE*).
cyclicsym_dcc2d4_360_loads.inp	Full model with all thermal loads (**FILM*, **CFILM*, **SFILM*, **CFLUX*, **DFLUX*, **DSFLUX*, **RADIATE*, **CRADIATE*, and **SRADIATE*).
cyclicsym_dc2d4_heat_nonlin.inp	Cyclic symmetry model with a nonlinear heat transfer analysis.
cyclicsym_dc2d4_heat_nonlin_360.inp	Full model corresponding to nonlinear heat transfer analysis.
cyclicsym_dc2d4_mis_heat_ee.inp	Cyclic symmetry model with a heat transfer analysis. Mismatched mesh with element-type secondary surface and element-type main surface.
cyclicsym_dc2d4_mis_heat_ne.inp	Cyclic symmetry model with a heat transfer analysis. Mismatched mesh with node-type secondary surface and element-type main surface.
cyclicsym_dc2d4_mis_360.inp	Full model corresponding to mismatched mesh of DC2D4 elements.
cyclicsym_dc2d8_heat2D.inp	Cyclic symmetry model with a heat transfer analysis using DC2D8 elements.
cyclicsym_dc2d8_heat2D_360.inp	Full model corresponding to the heat transfer analysis using DC2D8 elements.
cyclicsym_dc3d8_heat_CFLUX.inp	Cyclic symmetry model with a heat transfer analysis (**CFLUX*).
cyclicsym_dc3d8_heat_CFLUX_360.inp	Full model with a heat transfer analysis (**CFLUX*).
cyclicsym_dc3d8_heat_DFLUX.inp	Cyclic symmetry model with a heat transfer analysis (**DFLUX*).
cyclicsym_dc3d8_heat_DFLUX_360.inp	Full model with a heat transfer analysis (**DFLUX*).
cyclicsym_dc3d8_heat_DSFLUX.inp	Cyclic symmetry model with a heat transfer analysis (**DSFLUX*).
cyclicsym_dc3d8_heat_DSFLUX_360.inp	Full model with a heat transfer analysis (**DSFLUX*).
cyclicsym_dc3d8_heat_FILM.inp	Cyclic symmetry model with a heat transfer analysis (**FILM*).
cyclicsym_dc3d8_heat_FILM_360.inp	Full model with a heat transfer analysis (**FILM*).
cyclicsym_dc3d8_heat_RAD.inp	Cyclic symmetry model with a heat transfer analysis (**RADIATE*).

cyclicsym_dc3d8_heat_RAD_360.inp	Full model with a heat transfer analysis (**RADIATE*).
cyclicsym_dc3d8_heat_SFILM.inp	Cyclic symmetry model with a heat transfer analysis (**SFILM*).
cyclicsym_dc3d8_heat_SFILM_360.inp	Full model with a heat transfer analysis (**SFILM*).
cyclicsym_dc3d8_heat_SRAD.inp	Cyclic symmetry model with a heat transfer analysis (**SRADIATE*).
cyclicsym_dc3d8_heat_SRAD_360.inp	Full model with a heat transfer analysis (**SRADIATE*).
cyclicsym_dc3d8_heat_XYZ.inp	Cyclic symmetry model with a heat transfer analysis. The symmetry axis is a linear combination of the *X*, *Y*, and *Z* axes.
cyclicsym_dc3d8_heat_XYZ_360.inp	Full model with a heat transfer analysis. The symmetry axis is a linear combination of the *X*, *Y*, and *Z* axes.
cyclicsym_dc3d8_heat_axisY.inp	Cyclic symmetry model with a heat transfer analysis. The symmetry axis is parallel to the *Y*-axis.
cyclicsym_dc3d8_heat_axisY_360.inp	Full model with a heat transfer analysis. The symmetry axis is parallel to the *Y*-axis.

Abaqus/Aqua analysis

In this section:

- *Aqua load cases*
- *Jack-up foundation analysis*
- *Elastic-plastic joint elements*

Aqua load cases

Products: Abaqus/Standard Abaqus/Explicit Abaqus/Aqua

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Full submergence of structural members

Elements tested

B21 B21H B22 B22H B23 B23H B31 B31H B32 B32H B33 B33H
ELBOW31 ELBOW31B ELBOW31C ELBOW32
PIPE21 PIPE21H PIPE22 PIPE22H PIPE31 PIPE31H PIPE32 PIPE32H
RB2D2 RB3D2 R3D3 R3D4
T2D2 T2D2H T2D3 T2D3H T3D2 T3D2H T3D3 T3D3H

Problem description

The structural member (beam, pipe, elbow, or truss) is kept straight and constrained, and it is moved to different positions and orientations in different steps; where appropriate, it is given a uniform velocity and acceleration. The structural member is subjected to various drag and buoyancy loads in the different steps. The problems are described in detail in the input files. The concentrated and distributed load procedures are tested in these problems. The effective axial force (output variable ESF1) for beam, pipe, and truss elements is also tested.

The features and load types tested in each problem in the various steps are:

1. Buoyancy, PB.
2. Normal drag, static, FDD.
3. Tangential drag, static, FDT.
4. Normal drag, dynamic, FDD.
5. Tangential drag, dynamic, FDT.
6. Inertial drag, FI.
7. Normal drag, dynamic, partial immersion, FDD.
8. End-drag, dynamic, FD1, FD2.
9. End-drag, dynamic, TFD (concentrated load).
10. Inertial end-drag, FI1, FI2.
11. Inertial end-drag, TSI (concentrated load).
12. Transition-section buoyancy, TSB.
13. End-drag, dynamic, (additional test), FD1, FD2.
14. End-drag, dynamic, (additional test), TFD (concentrated load).
15. Wind-drag, dynamic, WDD.
16. Wind end-drag, dynamic, WD1, WD2.
17. Wind end-drag, dynamic, TWD (concentrated load).

The individual steps are named alphabetically as listed above. These names appear in the step headings.

Model:

Length	$10\sqrt{2}$
Orientation	45° with horizontal axis
Pipe section data	$r = 1.0$, $t = 0.05$

Material:

Young's modulus	30×10^9
Poisson's ratio	0.3

Aqua – environment:

Seabed elevation	0.0
Mean water elevation	40.0
Max. water elevation	40.0
Min. water elevation	40.0
Gravitational constant	32.2
Fluid mass density	1.987
Steady velocity specification: two-dimensional	
(v_x, v_y, elevation)	(2.0, 1.0, 0.0)
(v_x, v_y, elevation)	(2.0, 1.0, 2000.0)
Steady velocity specification: three-dimensional	
(v_x, v_z, elevation)	(2.0, 1.0, 0.0)
(v_x, v_z, elevation)	(2.0, 1.0, 2000.0)
(v_y = 0.0)	

Results and discussion

The correct total force can be determined analytically for the simple case of a straight structural member under drag or buoyancy loads, subjected to a uniform structural velocity or acceleration immersed in water with a constant velocity field. In all cases the reaction force at the beam nodes produced by Abaqus matches the analytical solution.

The analytically determined results are listed in the headings for each step in the input files.

Input files

eb22pxdb.inp	B21 elements.
eb2hpxdb.inp	B21H elements.
eb23pxdb.inp	B22 elements.
eb2ipxdb.inp	B22H elements.
eb2apxdb.inp	B23 elements.
eb2jpxdb.inp	B23H elements.

eb32pxdb.inp	B31 elements.
eb3hpxdb.inp	B31H elements.
eb33pxdb.inp	B32 elements.
eb3ipxdb.inp	B32H elements.
eb3apxdb.inp	B33 elements.
eb3jpxdb.inp	B33H elements.
exel1xdb.inp	ELBOW31 elements.
exelbxdb.inp	ELBOW31B elements.
exelbxdb.inp	ELBOW31C elements.
exel2xdb.inp	ELBOW32 elements.
ep22pxdb.inp	PIPE21 elements.
ep2hpxdb.inp	PIPE21H elements.
ep23pxdb.inp	PIPE22 elements.
ep2ipxdb.inp	PIPE22H elements.
ep32pxdb.inp	PIPE31 elements.
ep3hpxdb.inp	PIPE31H elements.
ep33pxdb.inp	PIPE32 elements.
ep3ipxdb.inp	PIPE32H elements.
er22sxdb.inp	RB2D2 elements.
er32sxdb.inp	RB3D2 elements.
er33sxdb.inp	R3D3 elements.
er34sxdb.inp	R3D4 elements.
et22sxdb.inp	T2D2 elements.
et2hsxdb.inp	T2D2H elements.
et23sxdb.inp	T2D3 elements.
et2isxdb.inp	T2D3H elements.
et32sxdb.inp	T3D2 elements.
et3hsxdb.inp	T3D2H elements.
et33sxdb.inp	T3D3 elements.
et3isxdb.inp	T3D3H elements.

Partial submergence of structural members

Elements tested

B21 B21H B22 B22H B23 B23H B31 B31H B32 B32H B33 B33H
ELBOW31C RB2D2 RB3D2
T2D2 T2D2H T2D3 T2D3H T3D2 T3D2H T3D3 T3D3H

Problem description

The structural member is positioned vertically in both the two- and three-dimensional cases, such that one-half of the structure is below the seabed and only the top half is subject to fluid loads.

Nodes of each element are constrained to a single node whose reaction force is monitored.

The features and load types tested in each problem in the various steps are:

1. Static analysis with drag load FDD and no wave loads.
2. Static analysis: dummy step to zero out the loads.
3. Dynamic analysis with inertial load FI.

Model:

Height of the structure	2
Section data	r = 1.0 for beams, A = 1.0 for trusses

Material:

Young's modulus	1×10^6

Aqua – environment:

Seabed elevation	0.0
Mean water elevation	2.0
Gravitational constant	32.2
Fluid mass density	1.99
Steady velocity specification: 2D/3D	
(v_x, v_y, v_z, elevation)	(1.0, 0.0, 0.0, 0.0)
(v_x, v_y, v_z, elevation)	(1.0, 0.0, 0.0, 2.0)

Airy wave parameters:

Amplitude	0.1
Period	10.0
Phase angle	0.0
Direction of travel	(1.0, 0.0)

Results and discussion

The results match the analytically determined reaction force.

Input files

eb22cxd1.inp	B21 elements.
ebxxcxd1.inp	All beam elements.
exelcxd1.inp	ELBOW31C elements.
etxxcxd1.inp	All truss elements.

Submergence of a rigid box

Elements tested

R3D3 R3D4

Problem description

A box composed of three-dimensional rigid elements is immersed in water subject to a buoyancy load (PB). The buoyancy forces and moments produced are measured by the reaction force at the rigid body reference node in four distinct configurations: in the initial configuration, as well as in the configurations produced when the body is given 60° of heel and then followed by 10° and 20° of trim.

Results and discussion

The Abaqus values for the buoyancy forces match the analytical values exactly. Because analytical values are not readily available at the moment, these values are compared with values produced by an independent code and agree to within one-quarter of 1%. The expected results are listed in the input files.

Input files

er33sxdb.inp	R3D3 elements.
er34sxdb.inp	R3D4 elements.

Eigenfrequency extraction with added mass

Elements tested

B21 T3D2

Problem description

Frequencies of natural vibration are computed for slender structures with different boundary conditions, with and without the effect of added mass.

Model:

Length	1000
Beam section data (circular)	$r = 3$

Material:

Young's modulus	4.32×10^9
Density	$\rho = 14.91$

Aqua – environment:

Seabed elevation	−100
Mean water elevation	100
Gravitational constant	32.2
Fluid mass density	2

Results and discussion

The analytically determined results and those given by Abaqus are listed at the top of each of the input files.

Input files

eb22cxd1.inp	Transverse vibration of simply supported beam.
eb22cxd2.inp	Transverse vibration of clamped-free cantilever beam.
eb22cxd3.inp	Longitudinal vibration of clamped-free cantilever beam.
et32pxdb.inp	Longitudinal vibration of clamped-free truss.

Spatial variation of steady current velocity

Elements tested

PIPE21 PIPE31

Problem description

Vertical structural members, fully submerged and constrained, are subjected to a steady current velocity that is uniform with respect to elevation but varies with position (x-coordinate for two-dimensional cases, and x- and y-coordinate for three-dimensional cases). The drag forces on the individual members can be determined analytically and compared to the nodal reaction forces.

The fluid velocity v_f is equal to 2.8961.

Model:

Height of the structure	10
Pipe section data	$r = 1.0$, $t = 0.05$

Material:

Young's modulus	30×10^6

Aqua – environment:

Seabed elevation	0.0
Mean water elevation	40.0
Gravitational constant	32.2
Fluid mass density	1.987

Steady velocity specification: two-dimensional case:

(v_x, v_y, v_z, x-coord.)	($1v_f$, 0.0, 0.0, 100.0)
(v_x, v_y, v_z, x-coord.)	($3v_f$, 0.0, 0.0, 300.0)
(v_x, v_y, v_z, x-coord.)	($3v_f$, 0.0, 0.0, 600.0)
(v_x, v_y, v_z, x-coord.)	($0v_f$, 0.0, 0.0, 900.0)

Steady velocity specification: three-dimensional case:

(v_x, v_y, v_z, x-coord., y-coord.)	($2v_f$, 0.0, 0.0, 100.0, 200.0)
(v_x, v_y, v_z, x-coord., y-coord.)	($6v_f$, 0.0, 0.0, 300.0, 200.0)
(v_x, v_y, v_z, x-coord., y-coord.)	($6v_f$, 0.0, 0.0, 600.0, 200.0)
(v_x, v_y, v_z, x-coord., y-coord.)	($0v_f$, 0.0, 0.0, 900.0, 200.0)
(v_x, v_y, v_z, x-coord., y-coord.)	($1v_f$, 0.0, 0.0, 100.0, 800.0)
(v_x, v_y, v_z, x-coord., y-coord.)	($3v_f$, 0.0, 0.0, 300.0, 800.0)
(v_x, v_y, v_z, x-coord., y-coord.)	($3v_f$, 0.0, 0.0, 600.0, 800.0)
(v_x, v_y, v_z, x-coord., y-coord.)	($0v_f$, 0.0, 0.0, 900.0, 800.0)

Results and discussion

The results match the analytically determined reaction forces at select locations.

Input files

ep22pxd5.inp	PIPE21 elements.
ep32pxd5.inp	PIPE31 elements.
ep22pxd5_xpl.inp	PIPE21 elements in Abaqus/Explicit.
ep32pxd5_xpl.inp	PIPE31 elements in Abaqus/Explicit.

Dynamic pressure, closed-end buoyancy loads

Elements tested

PIPE21 PIPE22 PIPE31

Problem description

This problem tests the dynamic pressure implementation and closed-end buoyancy loading for the three Abaqus/Aqua wave options. A vertical pile is fully constrained and subjected to buoyancy loading. The Airy, Stokes, and gridded wave options are used to calculate the total reaction force on the structure during a direct-integration implicit dynamic analysis procedure. Distributed load type PB is used with a 50-element model, and concentrated load type TSB is used with a one-element model.

Model:

Height of the structure	175.0 (100.0 below and 75.0 above mean water elevation)
Pipe section data	r = 1.0, t = 0.25

Material:

Young's modulus	1×10^6

Aqua – environment:

Seabed elevation	100.0
Mean water elevation	1100.0
Gravitational constant	32.2
Fluid mass density	2.0

Results and discussion

The results agree well with the analytically determined peak total reaction force.

Input files

ep32pxx1.inp	Airy waves, PIPE31 elements.
ep23pxx2.inp	Stokes waves, PIPE22 elements.
ep32pxx3.inp	Gridded wave data with linear interpolation, PIPE31 elements.
ep23pxx3.inp	Gridded wave data with quadratic interpolation, PIPE22 elements.
pb_airy_p31_xpl.inp	Airy waves, PIPE31 elements in Abaqus/Explicit.
pb_airy_p21_xpl.inp	Airy waves, PIPE21 elements in Abaqus/Explicit.

Gridded wave file

Problem description

This problem illustrates the creation of the gridded wave file. The unformatted binary gridded wave files used in *Dynamic pressure, closed-end buoyancy loads* (*ep32pxx3.inp* and *ep23pxx3.inp*) are created from ASCII format files containing the gridded wave data using a Fortran program.

Results and discussion

The files gridwave_3d.binary and gridwave_2d.binary are created for use in *Dynamic pressure, closed-end buoyancy loads*.

Input files

gridwave_2d.inp	ASCII format file containing two-dimensional gridded wave data.
gridwave_3d.inp	ASCII format file containing three-dimensional gridded wave data.
gridfile_2d.f	Fortran program to convert the two-dimensional ASCII data file to a binary gridded wave file.
gridfile_3d.f	Fortran program to convert the three-dimensional ASCII data file to a binary gridded wave file.

Miscellaneous partial submergence tests for Stokes waves

Elements tested

B21 PIPE21

Problem description

This problem tests the implementation of the effective axial force output quantity ESF1. Coincident, one-element, vertical piles are partially submerged in a Stokes wave field such that the element integration points change

between unsubmerged and submerged conditions during the analysis. The piles are fully constrained and subjected to distributed load type PB including internal fluid pressure. One pile is completely filled with internal fluid (Case A), and one is partially filled with internal fluid such that the element integration point is above the internal fluid free surface elevation (Case B). An amplitude variation is added to the distributed load definition in Cases A and B to produce, respectively, Cases C and D. Cases A and C use PIPE21 elements, and Cases B and D use B21 elements with general beam section to define the element properties. With the results from this analysis, the effective axial force output is tested using the postprocessing analysis procedure option.

Results and discussion

The effective axial force, ESF1, agrees with the analytical results for each case. The results are documented at the top of the *xesf1gen.inp* input file.

Input files

xesf1gen.inp	Input file for this analysis.
xesf1gep.inp	Input file that tests the postprocessing analysis procedure option.

Miscellaneous buoyancy loading

Elements tested

PIPE21

Problem description

This problem tests loading types PB and TSB when the fluid properties are prescribed as part of the loading. A general beam section procedure is used to describe the section properties.

Results and discussion

The results match the analytical solution.

Input files

pipepbtsb.inp	Input file for this analysis.

Jack-up foundation analysis

Products: Abaqus/Standard Abaqus/Aqua

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Initial embedment analysis

Elements tested

JOINT2D JOINT3D

Problem description

The initial embedment calculation as a function of the preload is verified for sand and clay models. A two-step single-element elastic analysis is performed with a given jack-up foundation preload for the different models. JOINT3D elements are used. In the first step the base node is fixed, and the tip node is subjected to concentrated forces and moments. The second step is a static perturbation analysis about the previous step. The analysis is done for the six models described below. It is verified that the embedment value is correct and that the elastic modulus has the correct dependence on embedment.

Force units are kN, and length units are meters.

1. Sand model, cylindrical spud can:

Spud can diameter	10.9
Spud can cone angle	180°
Foundation preload	50600
Foundation tensile capacity	0.0
Soil submerged unit weight	10.0
Soil friction angle	33°
Soil Poisson's ratio	0.2
Foundation elastic shear moduli, G_{vv}	5.14×10^4
G_{hh}	3.87×10^3
G_{rr}	2.04×10^4
Constant coefficient, Λ_1	1.0
Constant coefficient, Λ_2	0.5

2. Sand model, conical spud can—embedment greater than critical:

The properties for the soil are the same as in Case a.

Spud can diameter	10.9
Spud can cone angle	150°
Foundation preload	50,000

Foundation tensile capacity	0

3. Sand model, conical spud can—embedment less than critical:

   The properties of the spud can and the soil are same as in Case b. The foundation preload is 15000 for this case.

4. Clay model, cylindrical spud can:

Spud can diameter	20.0
Spud can cone angle	180°
Foundation preload	1.3×10^5
Foundation tensile capacity	0.0
Soil submerged unit weight	10.0
Soil undrained shear strength	150.0
Soil Poisson's ratio	0.5
Foundation elastic shear moduli, G_{vv}	1.56×10^4
G_{hh}	2.34×10^3
G_{rr}	6.38×10^4
Hardening parameter, *a*	7.204×10^4
Hardening parameter, *b*	1.978×10^3

5. Clay model, conical spud can—embedment greater than critical:

Spud can diameter	20.0
Spud can cone angle	150°
Foundation preload	8.5×10^5
Foundation tensile capacity	0.0
Soil submerged unit weight	10.0
Soil undrained shear strength	50.0
Soil Poisson's ratio	0.5
Foundation elastic shear moduli, G_{vv}	1.56×10^4
G_{hh}	2.34×10^3
G_{rr}	6.38×10^4
Hardening parameter, *a*	-2.395×10^{-5}
Hardening parameter, *b*	8.777×10^{-6}
Hardening parameter, *c*	2.9294

6. Clay model, conical spud can—embedment less than critical:

   The properties of the soil and the spud can are the same as in Case e. The foundation preload is 1.3×10^5.

General: Six additional elements test initial field variable dependence of the material properties. At the specified values of the field variables these elements have the properties of models a, b, c, d, e, and f.

Results and discussion

The initial embedment for each of the models is in agreement with analytical results.

Input files

paqajembed.inp	Initial embedment analysis.

Push-over analysis: sand model

Problem description

The structure tested is a four-leg square platform with a footing at each leg corner. The model can be reduced to two dimensions because of symmetry. The model is projected onto a vertical plane that cuts diagonally across the platform. The legs are modeled with B21 beam elements, and the foundation is modeled with JOINT2D elements. The platform is modeled as a two-dimensional portal frame, with one windward leg, one leeward leg, and two legs in the middle. The platform is considered rigid and is modeled with RB2D2 elements. Four push-over analyses with different foundation bearing capacities are performed.

Force units are kN, and length units are meters.

Leg length	59
Leg EI	1.0×10^{15}
Leg AE	3.0×10^{15}
Leg GA	2.0×10^{15}
Horizontal distance from platform c.g. to leeward leg	29.33
Horizontal distance from platform c.g. to windward leg	29.33
Horizontal distance from platform c.g. to middle legs	0
Spud can diameter	14.0
Spud can cone angle	180°
Foundation preload, four cases	387500, 530000, 650000, 775000
Foundation tensile capacity	40000
Spud can initial vertical load	52250
Vertical distance from c.g. to load application point	0
Soil submerged unit weight	10.0
Soil friction angle	35°
Soil Poisson's ratio	0.2
Foundation elastic shear moduli, G_{vv}	1.63×10^5
G_{hh}	2.92×10^4
G_{rr}	2.10×10^4
Constant coefficient, Λ_1	0.3
Constant coefficient, Λ_2	0.3

The ultimate bearing capacity is determined by applying a load larger than the bearing capacity in a static step with a time period of 1. This load ramps up over the step, and the analysis fails to converge when the bearing

capacity is reached. The capacity is determined by multiplying the reference load (in these cases 200000 kN) by the fraction of the time step completed.

For accurate results in a push-over analysis, experience shows that small time increments should be used to integrate the plasticity equations accurately. These analyses were each run with three different fixed time increments.

Results and discussion

The ultimate bearing capacity for the four cases of foundation preloads are found to be in good agreement with the following reference capacities calculated using an external code.

	Ref.	Abaqus capacity		
Preload	capacity	$\Delta t = 1 \times 10^{-2}$	$\Delta t = 1 \times 10^{-3}$	$\Delta t = 1 \times 10^{-4}$
387.5×10^3	126×10^3	30×10^3	125×10^3	124×10^3
530×10^3	137×10^3	140×10^3	136×10^3	136×10^3
650×10^3	145×10^3	146×10^3	143×10^3	143×10^3
775×10^3	150×10^3	152×10^3	153×10^3	150×10^3

The input file *paqajsandp.inp* models the 775000 kN preload case, with an applied force of 95% of the ultimate capacity of 150000 kN over a step of 100 increments.

Input files

paqajsandp.inp Push-over analysis for sand model.

Monotonic loading analysis: clay model

Problem description

The test problem is a monotonic horizontal loading analysis of a triangular three-leg jack-up rig on clay. The rig is modeled as a frame composed of rigid elements, with two windward legs and one leeward leg. For the two-dimensional analysis the model is projected on a vertical plane of symmetry. Loading for both the two- and three-dimensional analyses is in this plane, so both analyses produce the same results. The loading consists of an applied horizontal load at a point below the rigid frame. The legs are modeled with B21 elements, and the joints are modeled with JOINT2D elements.

The properties of the soil and the spud can are as described in Case d of the initial embedment analysis.

Leg length	110.6
Leg EI	2.48×10^9
Leg AE	1.54×10^{15}
Leg GA	3.61×10^{15}
Horizontal distance from platform c.g. to leeward leg	37.0
Horizontal distance from platform c.g. to windward leg	18.0
Spud can initial vertical load	6.6×10^4
Vertical distance from c.g. to load application point	−55.0

Results and discussion

The estimated load paths for the windward and the leeward legs are in agreement with the load paths calculated from an external code.

Input files

paqajclaym.inp	Monotonic loading analysis for clay model.
paqajclaym3d.inp	Monotonic loading analysis for clay model, three-dimensional.

Clay model with conical spud can

Problem description

The test structure is the same as that of *Monotonic loading analysis: clay model*. The soil plastic properties are different, and the spud can is conical. A conical spud can produces rather different results in this case, even in the elastic region, and the model verifies that the elastic properties depend correctly on the plastic properties through the embedment. The analysis consists of horizontal loading of the rig up to the value of 18000 kN.

The soil and spud can properties are as given in Case e of the initial embedment analysis. The rig dimensions are the same as that of the monotonic loading analysis.

Results and discussion

The load-displacement curve for the centroid of the platform and the moment-horizontal load curves for the windward and leeward spud cans are compared to those obtained from an external code and are in agreement.

Input files

paqajclaymc.inp	Monotonic loading analysis for clay model with conical spud can.

Monotonic loading: member

Problem description

The test structure is a half-model of a four-leg square rig, projected on the vertical, nondiagonal plane of symmetry. The horizontal and vertical loads are applied at the center of gravity of the platform. The shear stiffness of the legs is not included in the model; B23 elements are used. The spud cans are modeled as elastic-perfectly plastic in this case, using the “member”-type plasticity model. The vertical load is ramped up from 20 to 100 in the first step and then held constant until the end of the step. In the next step the horizontal load is ramped to 14.

The dimensions of the rig in the plane, the beam properties, and the elastic properties of the spud can are as given in the clay push-over analysis. The plastic properties of the member are given below:

Parabolic yield function parameters:

V_c	100
V_t	100
M_u	2400
H_u	10^{15}

Results and discussion

The moment-axial interaction for the leeward spud can and the member is calculated, plotted, and determined to be proceeding correctly along the fixed yield surface.

Input files

paqajmembm.inp	Monotonic loading for perfectly plastic “member” model.

Elastic-plastic joint elements

Products: Abaqus/Standard Abaqus/Aqua

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

JOINT2D JOINT3D

Problem description

A four-step single-element test is performed for two-dimensional and three-dimensional joint elements. The tests include conical and cylindrical cross-sections, with both diagonal and fully populated elastic stiffness material cases. The behavior of the joint elements is defined in a local coordinate system with the results output in the same coordinate system.

Seven different spud can models are used:

1. Two-dimensional cylindrical spud can, $D = 1.6$, with general moduli, $k_{1111} = 2000$, $k_{1122} = -1000$, $k_{2222} = 3000$, $k_{1112} = -2000$, $k_{2212} = 0.0$, $k_{1212} = 6000$.
2. Two-dimensional cylindrical spud can, $D = 1.25$, with spud can moduli $G_{vv} = 840.0$, $G_{hh} = 1643.0$, $G_{rr} = 2150.4$, Poisson's ratio, $\nu = 0.3$.
3. Two-dimensional conical spud can, $D = 1.25$, $\theta = 60°$ with spud can moduli and Poisson's ratio as in Case b, an initial embedment of 0.5 m (less than critical embedment).
4. Two-dimensional conical spud can, $D = 1.25$, $\theta = 60°$ with spud can moduli and Poisson's ratio as in Case b, an initial embedment of 2.5 m (greater than critical embedment).
5. Three-dimensional cylindrical spud can, $D = 1.1$, with general moduli, $k_{1111} = 1000$, $k_{1122} = 0.0$, $k_{2222} = 2000$, $k_{1133} = 0.0$, $k_{2233} = -1200$, $k_{3333} = 3000$, $k_{1112} = 0.0$, $k_{2212} = 0.0$, $k_{3312} = 0.0$, $k_{1212} = 5000$, $k_{1113} = 0.0$, $k_{2213} = 0.0$, $k_{3313} = 1000$, $k_{1213} = 0.0$, $k_{1313} = 6000$, $k_{1123} = 0.0$, $k_{2223} = 1000$, $k_{3323} = 0.0$, $k_{1223} = 0.0$, $k_{1323} = 0.0$, $k_{2323} = 2000$.
6. Three-dimensional cylindrical spud can, $D = 1.5$, with spud can moduli, $G_{vv} = 700$, $G_{hh} = 1095.2$, $G_{rr} = 4666.3$, torsional elastic spring stiffness $k_t = 5000$, Poisson's ratio, $\nu = 0.3$.
7. Three-dimensional conical spud can, $D = 1.5$, $\theta = 60°$, with spud can moduli $G_{vv} = 202.1$, $G_{hh} = 474.3$, $G_{rr} = 176.83$, torsional elastic spring stiffness $k_t = 4500$, Poisson's ratio, $\nu = 0.3$, $D = 1.5$, initial embedment = 0.321 (less than critical).

Four additional elements test field variable dependence of the material properties. At the specified values of the field variables, these elements have the properties of models a, b, e, and f.

Boundary conditions and loading: In the first step both the base node and the tip node are subjected to prescribed displacements and rotations. In the second step the previous boundary conditions are removed, and the base node is displaced by prescribing displacements and rotations. The tip node is free to move and should

follow the base node for this case. In the third step the base node is fixed, and the tip node is subjected to concentrated forces and moments. The fourth step is a perturbation step about the previous step, with loads perturbed by 50% of those in the previous general step.

Results and discussion

The results obtained match the analytical results.

Input files

exepxlx1.inp	Linear elastic tests for elastic-plastic joint elements.

Design sensitivity analysis

In this section:

- *Direct design sensitivity analysis*
- *Adjoint design sensitivity analysis*

Direct design sensitivity analysis

In this section:

- *Design sensitivity analysis for continuum elements*
- *Design sensitivity analysis for structural elements*
- *Design sensitivity analysis for special-purpose elements*
- *Design sensitivity analysis for elastic materials*
- *Design sensitivity analysis for contact*
- *Design sensitivity analysis controls*

Design sensitivity analysis for continuum elements

Products: Abaqus/Design

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D4 C3D4H C3D6 C3D6H C3D8 C3D8H C3D8I C3D8IH C3D8R C3D8RH C3D10 C3D10H C3D10M C3D10MH C3D15 C3D15H C3D15V C3D15VH C3D20 C3D20H C3D20R C3D27 C3D27H C3D27R C3D27RH
CAX3 CAX3H CAX4 CAX4H CAX4I CAX4IH CAX4R CAX4RH CAX6 CAX6H CAX6M CAX6MH CAX8 CAX8H CAX8R CAX8RH
CGAX3 CGAX3H CGAX4 CGAX4H CGAX4R CGAX4RH CGAX6 CGAX6H CGAX8 CGAX8H CGAX8R CGAX8RH
CPE3 CPE3H CPE4 CPE4H CPE4H CPE4I CPE4IH CPE4R CPE4RH CPE6 CPE6H CPE6M CPE6MH CPE8 CPE8H CPE8R CPE8RH
CPEG3 CPEG3H CPEG4 CPEG4H CPEG4I CPEG4IH CPEG4R CPEG4RH CPEG6 CPEG8 CPEG8H CPEG8R CPEG8RH
CPS3 CPS4 CPS4I CPS4R CPS6 CPS6M CPS8 CPS8R

Features tested

This section includes a general set of simple tests to verify the design sensitivity analysis (DSA) technique for stress/displacement continuum elements for static steps. Geometrically linear and nonlinear tests are done for both total and incremental DSA formulations. In addition, selected problems also test static perturbation steps and frequency steps. A full range of design parameters is used, including those related to sizing (for example, material properties, thickness) and shape (that is, nodal coordinates). The results verified are primarily displacement sensitivities for static steps and eigenvalue sensitivities for frequency steps.

Problem description

All problems are one- or two-element models with elastic or hyperelastic material properties. The models are fixed at one end and loaded using displacements, point loads, or distributed loads at the opposite end. At least one material property and one nodal coordinate are used as design parameters for each test; a sizing parameter, such as thickness, is also used as a design parameter if appropriate for the particular model.

Results and discussion

All sensitivity results are verified by comparison to hand calculations or to overall finite difference results.

Input files

pdsatotcpe.inp	Total DSA with plane strain stress/displacement elements.
pdsainccpe.inp	Incremental DSA with plane strain stress/displacement elements.
pdsatotcps.inp	Total DSA with plane stress stress/displacement elements.
pdsainccps.inp	Incremental DSA with plane stress stress/displacement elements.
pdsatotcpeg.inp	Total DSA with generalized plane strain stress/displacement elements.
pdsainccpeg.inp	Incremental DSA with generalized plane strain stress/displacement elements.

pdsatotc3d.inp	Total DSA with three-dimensional stress/displacement continuum elements.
pdsaincc3d.inp	Incremental DSA with three-dimensional stress/displacement continuum elements; includes frequency step.
pdsatotcax.inp	Total DSA with axisymmetric stress/displacement continuum elements; includes frequency step.
pdsainccax.inp	Incremental DSA with axisymmetric stress/displacement continuum elements.
pdsatotcgax.inp	Total DSA with axisymmetric stress/displacement elements with twist.
pdsainccgax.inp	Incremental DSA with axisymmetric stress/displacement elements with twist.

Design sensitivity analysis for structural elements

Products: Abaqus/Design

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

B21 B21H B22 B22H B23 B23H B31 B31H B31OS B31OSH
B32 B32H B32OS B32OSH B33 B33H
M3D3 M3D4 M3D4R M3D6 M3D8 M3D8R M3D9 M3D9R
MAX1 MAX2 MGAX1 MGAX2
S4R S4R5 S4 S3R STRI3 S8R S8R5 S9R5 STRI65
SAX1 SAX2 SAXA14 SAXA24
T2D2 T2D2H T2D3 T2D3H T3D2 T3D2H T3D3 T3D3H

Features tested

This section includes a general set of simple tests to verify the design sensitivity analysis (DSA) technique for membrane and shell elements. Geometrically linear and nonlinear tests are done for both total and incremental DSA formulations. A full range of design parameters is used, including those related to sizing (for example, material properties, thickness) and shape (that is, nodal coordinates). All problems test static steps, and some selected problems also test frequency steps.

Problem description

All problems are two-element models with elastic or composite material properties. The models are fixed at one end and loaded using displacements, point loads, or distributed loads at the opposite end. At least one material property and one nodal coordinate are used as design parameters for each test; a sizing parameter, such as thickness, is also used as a design parameter if appropriate for the particular model.

Results and discussion

All sensitivity results are verified by comparison to hand calculations or to overall finite difference results.

Input files

pdsatottruss.inp	Total DSA with truss elements.
pdsainctruss.inp	Incremental DSA with truss elements; includes frequency step.
pdsatotm3d.inp	Total DSA with membrane elements.
pdsaincm3d.inp	Incremental DSA with membrane elements; includes frequency step.
pdsatotmax.inp	Total DSA with axisymmetric membrane elements.
pdsaincmax.inp	Incremental DSA with axisymmetric membrane elements.
pdsatotmaxa.inp	Total DSA with axisymmetric membrane elements with twist.
pdsaincmaxa.inp	Incremental DSA with axisymmetric membrane elements with twist.
pdsaincbeam.inp	Incremental DSA with beam elements.
pdsatots3d.inp	Total DSA with shell elements; includes frequency step.
pdsaincs3d.inp	Incremental DSA with shell elements; includes frequency step.

pdsatotsax.inp	Total DSA with axisymmetric shell elements; includes frequency step.
pdsaincsax.inp	Incremental DSA with axisymmetric shell elements.
pdsatotsaxa.inp	Total DSA with axisymmetric shell elements with asymmetric deformations.
pdsaincsaxa.inp	Incremental DSA with axisymmetric shell elements with asymmetric deformations.
pdsatotbeam.inp	Total DSA with beam elements.

Design sensitivity analysis for special-purpose elements

Products: Abaqus/Design

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

GK2D2 GK2D2N GKPS4 GKPS4N GKPS6 GKPS6N GKAX2 GKAX2N GKAX4 GKAX4N GKAX6 GKAX6N GKPE4 GKPE6
GK3D2 GK3D2N GK3D4L GK3D4LN GK3D8 GK3D8N GK3D6 GK3D6N GK3D18 GK3D12M GK3D18N GK3D12MN

Features tested

This section includes a general set of simple tests to verify the design sensitivity analysis (DSA) technique for gasket elements. Geometrically nonlinear tests are done for both total and incremental DSA formulations.

Problem description

All problems are static problems with gaskets sandwiched between continuum elements. The design parameters chosen govern the gasket section properties.

Results and discussion

The results are verified by comparing them with the results from the overall finite difference method.

Input files

pdsainctwogasket.inp	Incremental DSA with two-dimensional gasket elements.
pdsatottwogasket.inp	Total DSA with two-dimensional gasket elements.
pdsaincthreegasket.inp	Incremental DSA with three-dimensional gasket elements.
pdsatotthreegasket.inp	Total DSA with three-dimensional gasket elements.

Design sensitivity analysis for elastic materials

Products: Abaqus/Design

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

CPE4I CAX4 C3D8 S4R

Features tested

This section includes simple tests to verify DSA for the isotropic elasticity, hyperelasticity (Ogden and polynomial models), and hyperfoam material options. The elastic material models are tested as geometrically linear cases that include temperature dependence. The hyperelastic models are tested as geometrically nonlinear cases with the material properties input as coefficients (no test data input). The material coefficients are chosen as the design parameters. For all problems sensitivities of element and node responses are verified for static steps, and for selected problems sensitivities of eigenvalues and eigenfrequencies are verified for frequency steps.

Problem description

The tests are performed on a square or a cylindrical block discretized with four to eight elements. The block is held fixed at one end and loaded using prescribed displacements or point loads at the other end. Key material coefficients used in defining the material models are the primary design parameters, while some shape parameters are made design parameters as appropriate.

Results and discussion

All response sensitivities are verified by comparison to overall finite difference results.

Input files

mdsatotaxel.inp	Total DSA, elastic axisymmetric model.
mdsatotaxelt.inp	Total DSA, elastic axisymmetric model with temperature dependence.
mdsatot3del.inp	Total DSA, elastic three-dimensional model.
mdsatot3delt.inp	Total DSA, elastic three-dimensional model with temperature dependence.
mdsainc2dhyp.inp	Incremental DSA, hyperelastic (polynomial) two-dimensional model.
mdsainc3dhyp.inp	Incremental DSA, hyperelastic (polynomial) three-dimensional model.
mdsainc2dhyo.inp	Incremental DSA, hyperelastic (Ogden) two-dimensional model.
mdsainc3dhyo.inp	Incremental DSA, hyperelastic (Ogden) three-dimensional model.
mdsainc2dhyf.inp	Incremental DSA, hyperfoam two-dimensional model; includes frequency step.
mdsainc3dhyf.inp	Incremental DSA, hyperfoam three-dimensional model.
mdsaoritrs.inp	Incremental DSA, elastic with engineering constant model, shell element with orientation.

Design sensitivity analysis for contact

Products: Abaqus/Design

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D8 C3D10M CPE4 CPE6M S4R

Features tested

This section includes a set of simple tests to verify DSA for contact between solid displacement elements and rigid surfaces with small-sliding and finite-sliding surface interaction. Both analytical and discrete rigid surfaces are used. The interaction between the rigid and deformable surfaces is assumed to be frictionless for all small-sliding surface interactions. Isotropic Coulomb friction with a friction coefficient of 0.2 is assumed for the finite-sliding tests. Two-dimensional and three-dimensional first-order solids with hyperelastic material models are tested. Shape parameters that affect the secondary surface and friction coefficient are chosen as the design parameters, and the sensitivities of the contact responses CPRESS and CDISP are verified.

Problem description

The tests are performed on a square block discretized with four to eight elements. The structure is held fixed at one end, and a rigid die is pushed onto the other end using prescribed displacements. The incremental DSA formulation is used in all tests. Shape parameters that change the shape of the secondary surface are chosen as the primary design parameters.

Results and discussion

The response sensitivities are verified by comparison to overall finite difference results.

Input files

idsaincsm2d_arig.inp	Incremental DSA, small-sliding, two-dimensional model, analytical rigid surface.
idsaincsm2d_arig_surf.inp	Incremental DSA, small-sliding, two-dimensional model, analytical rigid surface, surface-to-surface constraint enforcement method.
idsaincsm2d_drig.inp	Incremental DSA, small-sliding, two-dimensional model, discrete rigid surface.
idsaincsm2d_drig_surf.inp	Incremental DSA, small-sliding, two-dimensional model, discrete rigid surface, surface-to-surface constraint enforcement method.
idsaincsm2dm_arig.inp	Incremental DSA, small-sliding, two-dimensional model, modified triangles, analytical rigid surface.
idsaincsm2dm_arig_surf.inp	Incremental DSA, small-sliding, two-dimensional model, modified triangles, analytical rigid surface, surface-to-surface constraint enforcement method.
idsaincsm3d_arig.inp	Incremental DSA, small-sliding, three-dimensional model, analytical rigid surface.
idsaincsm3d_arig_surf.inp	Incremental DSA, small-sliding, three-dimensional model, analytical rigid surface, surface-to-surface constraint enforcement method.

idsaincsm3d_drig.inp	Incremental DSA, small-sliding, three-dimensional model, discrete rigid surface.
idsaincsm3d_drig_surf.inp	Incremental DSA, small-sliding, three-dimensional model, discrete rigid surface, surface-to-surface constraint enforcement method.
idsaincsm3dm_arig.inp	Incremental DSA, small-sliding, three-dimensional model, modified tetrahedra, analytical rigid surfaces.
idsaincsm3dm_arig_surf.inp	Incremental DSA, small-sliding, three-dimensional model, modified tetrahedra, analytical rigid surfaces, surface-to-surface constraint enforcement method.
idsainc2d_arig.inp	Incremental DSA, finite-sliding, two-dimensional model, analytical rigid surface with friction.
idsatot2d_arig.inp	Total DSA, finite-sliding, two-dimensional model, analytical rigid surface with friction.
idsainc2d_drig.inp	Incremental DSA, finite-sliding, two-dimensional model, discrete rigid surface with friction.
idsatot2d_drig.inp	Total DSA, finite-sliding, two-dimensional model, discrete rigid surface with friction.
idsainc3d_arig.inp	Incremental DSA, finite-sliding, three-dimensional model, analytical rigid surface with friction.
idsatot3d_arig.inp	Total DSA, finite-sliding, three-dimensional model, analytical rigid surface.
idsainc3d_drig.inp	Incremental DSA, finite-sliding, three-dimensional model, discrete rigid surface with friction.
idsatot3d_drig.inp	Total DSA, finite-sliding, three-dimensional model, discrete rigid surface.
idsaincshell.inp	Incremental DSA, finite-sliding, structural model, analytical rigid surface with friction.

Design sensitivity analysis controls

Products: Abaqus/Design

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Features tested

This section includes various tests used to verify the behavior of the DSA controls. Output variables, unsupported elements, and restart are also verified.

Problem description

Elements are subjected to concentrated or distributed loads. Static analyses are performed.

Results and discussion

The response sensitivities are verified by comparison to overall finite difference results.

Input files

cdsatotpar.inp	Total DSA testing user perturbation size control.
cdsaincpar.inp	Incremental DSA testing user perturbation size, gravity loading, and mass sensitivity controls.
cdsatottol.inp	Total DSA testing sizing frequency and tolerance controls.
cdsainctol.inp	Incremental DSA testing sizing frequency and tolerance controls.
cdsaunspele.inp	Incremental DSA testing unsupported elements (beam, spring, gasket, and connector elements with the supported C3D8 element).
mdsaoritrs_static_restart.inp	Restart static analysis with DSA.
mdsaoritrs_frequency_restart.inp	Restart frequency analysis with DSA.
cdsaincpar_restart.inp	Restart nonlinear frequency analysis with DSA.
cdsaincload.inp	Incremental DSA with design-dependent concentrated loads.
cdsatotload.inp	Total DSA with design-dependent concentrated loads.

Adjoint design sensitivity analysis

In this section:

- *Adjoint design sensitivity analysis for static procedures*
- *Adjoint design sensitivity analysis for frequency procedures*
- *Adjoint design sensitivity analysis for implicit transient dynamic procedures*
- *Adjoint design sensitivity analysis for combined Abaqus and fe-safe–based fatigue procedures*

Adjoint design sensitivity analysis for static procedures

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D4H C3D6H C3D8 C3D8H C3D8R C3D10H C3D10HS C3D10MH C3D15H C3D20H

CPE4H CPE4RH CPE6H CPE6M CPE6MH CPE8H CPE8R CPE8RH
CAX4 CAX4H CAX6 CAX6H CAX6M CAX6MH CAX8 CAX8H CAX8R
CGAX3 CGAX3H CGAX4H CGAX4 CGAX6 CGAX6H CGAX8 CGAX8H CGAX8R

S3 S4 S4R S8R

Features tested

This section includes a general set of simple tests to verify the adjoint design sensitivity analysis techniques for shell and solid continuum element types for a static procedure. The tests include general steps that are either geometrically linear or nonlinearly modeled, both with and without constitutive material nonlinearities. Linear perturbation steps with multiple load cases are also tested. A full range of design variables are used, including those related to topology, sizing, ABD stiffness, and shape/bead sensitivities. A number of different design responses, including user-defined element design responses based on either the stress or the plastic strain, are also tested.

Problem description

All tests are either one or four element models with elastic or elastic-plastic constitutive material properties. The models are loaded using either concentrated loads, distributed body loads, or prescribed temperature. The design parameters are one of the following: shell element thickness (for sizing sensitivities); composite shell element layer thickness and orientation angle; ABD stiffness (for shell general sections); nodal position (for shape/bead sensitivities); or material relative density that scales the constitutive material properties, physical density, and stress relaxation (for topology sensitivities).

Results and discussion

All adjoint sensitivity results are verified by comparing to results obtained using finite difference calculations.

Input files

S4_static_DV_Node.inp	Bead sensitivities for S4 elements with nodal positions as design variables.
S4_static_DV_Node_Oper.inp	Bead sensitivities for S4 elements with nodal positions as design variables. Uses operators in defining node and element design responses.
S4_static_DV_Thick.inp	Sizing sensitivities for S4 elements with shell thickness as design variables.
S4_static_DV_Thick_Oper.inp	Sizing sensitivities for S4 elements with shell thickness as design variables. Uses operators in defining node and element design responses.
S4_static_DV_Stiff.inp	Topology sensitivities for S4 elements with relative material density as design variables.

S4_static_DV_Stiff_Oper.inp	Topology sensitivities for S4 elements with relative material density as design variables. Uses operators in defining node and element design responses.
C3D10HS_static_LC_DV_Node.inp	Shape sensitivities for a multiple load case analysis with C3D10HS elements; nodal position as design variables.
C3D10HS_static_LC_DV_Stiff.inp	Topology sensitivities for a multiple load case analysis with C3D10HS elements; relative material density as design variables.
S4R_static_DV_Node_User_S.inp	Bead sensitivities for S4R elements with nodal position as design variables. Stress-based user-defined element response defined using user subroutine *UELEMDRESP*.
S4R_static_DV_Stiff_User_S.inp	Topology sensitivities for S4R elements with relative material density as design variables. Stress-based user-defined element response defined using user subroutine *UELEMDRESP*.
S4R_static_DV_Thick_User_S.inp	Sizing sensitivities for S4R elements with shell thickness as design variables. Stress-based user-defined element response defined using user subroutine *UELEMDRESP*.
Uelemdresp_S.f	User subroutine *UELEMDRESP* to define stress-based element design response.
S4R_static_DV_Layer_Thick_Angle_User.inp	Sizing sensitivities for S4R and S3 elements with composite shell layer thickness and orientation angle as design variables. The user-defined element response, Tsai-Wu and Hoffman failure criteria, is defined using user subroutine *UELEMDRESP*.
Uelemdresp_FailStress.f	User subroutine *UELEMDRESP* to define Tsai-Wu and Hoffman failure criteria for element design response.
C3D8R_static_DV_Node_User_PE.inp	Shape sensitivities for C3D8R elements with nodal position as design variables. Plastic strain–based user-defined element response defined using user subroutine *UELEMDRESP*.
C3D8R_static_DV_Stiff_User_PE.inp	Topology sensitivities for C3D8R elements with relative material density as design variables. Plastic strain–based user-defined element response defined using user subroutine *UELEMDRESP*.
Uelemdresp_PE.f	User subroutine *UELEMDRESP* to define plastic strain–based element design response.
C3D8_static_DV_Node_SINV.inp	Shape sensitivities of principal stresses and signed von Mises stress responses for C3D8 elements with nodal positions as design variables.
C3D8_static_DV_Stiff_SINV.inp	Topology sensitivities of principal stresses and signed von Mises stress responses for C3D8 elements with relative material density as design variables.
S4R_static_DV_Stiff_Neuber_Table.inp	Topology sensitivities of Neuber plasticity correction factors–based design responses for S4R elements with relative material density as design variables. The material data for the plasticity corrections are defined in tabular form.
S4R_static_DV_Node_Neuber_Table.inp	Bead sensitivities of Neuber plasticity correction factors–based design responses for S4R elements with nodal positions as design variables. The material data for the plasticity corrections are defined in tabular form.
S4R_static_DV_Thick_Neuber_Table.inp	Sizing sensitivities of Neuber plasticity correction factors–based design responses for S4R elements with shell thickness as design variables. The material data for the plasticity corrections are defined in tabular form.
S4R_static_DV_Stiff_NeuberGlinka_RO.inp	Topology sensitivities of Neuber and Glinka plasticity correction factors–based design responses for S4R elements with material relative density as design variables. The material data for the plasticity corrections are defined using the Ramberg-Osgood model.
S4R_static_DV_Node_NeuberGlinka_RO.inp	Bead sensitivities of Neuber and Glinka plasticity correction factors–based design responses for S4R elements with nodal positions as design variables.

	The material data for the plasticity corrections are defined using the Ramberg-Osgood model.
S4R_static_DV_Thick_ NeuberGlinka_RO.inp	Sizing sensitivities of Neuber and Glinka plasticity correction factors–based design responses for S4R elements with shell thickness as design variables. The material data for the plasticity corrections are defined using the Ramberg-Osgood model.
C2D_static_DV_Stiff_H_M.inp	Topology sensitivities of various design responses for a number of 2D elements with regular, hybrid, modified, and modified-hybrid formulations, with material relative density as design variables.
CAX_static_DV_Stiff_H_M.inp	Topology sensitivities of various design responses for a number of axisymmetric elements with regular (that is, nonhybrid or nonmodified), hybrid, modified, and modified hybrid formulations, with material relative density as design variables.
CGAX_static_DV_Stiff_H.inp	Topology sensitivities of various design responses for a number of 2D axisymmetric elements with twist, with material relative density as design variables. Both regular and hybrid elements were tested.
C3D_static_DV_Stiff_H_M.inp	Topology sensitivities of various design responses for a number of 3D hybrid and modified elements, with material relative density as design variables.
C3D10MH_static_DV_Node_ Hyperelas.inp	Shape sensitivities of various design responses for a C3D10MH element with hyperelastic material behavior, with nodal positions as design variables.
CPE6MH_static_DV_Node_Pert.inp	Shape sensitivities of various design responses for a CPE6MH element in a linear perturbation procedure, with nodal positions as design variables.
CAX6H_static_DV_Node_ Linear.inp	Shape sensitivities of various design responses for a CAX6H element in a linear procedure, with nodal positions as design variables.
S3_S4_S4R_S8R_General_Section_ static_DV_Stiffess_Linear.inp	ABD stiffness sensitivities of various design responses for elements S3, S4, S4R, and S8R defined using shell general sections in a linear procedure, with 21 ABD stiffness components as design variables.

Adjoint design sensitivity analysis for frequency procedures

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

B31 C3D8

Features tested

This section includes a general set of simple tests to verify the adjoint design sensitivity analysis techniques for beam and solid continuum element types for a frequency step. A full range of design variables are used, including those related to the topology relative material density, the lattice beam element radius, and nodal shape sensitivities. Sensitivities for multiple eigenfrequencies, as well as for models including special features such as model change, are tested.

Problem description

All tests use constitutive elastic material properties. The design parameters are either the relative material density, beam element radius, or nodal positions.

Results and discussion

All adjoint sensitivity results are verified by comparing them to results obtained using finite difference calculations.

Input files

B31_eigen_DV_Radius.inp	Lattice sensitivities for B31 elements with beam radius as design variables.
B31_eigen_DV_Stiff.inp	Topology sensitivities for B31 elements with relative material density as design variables.
C3D8_eigen_DV_Node.inp	Shape sensitivities for C3D8 elements with nodal position as design variables.
C3D8_eigen_DV_Stiff.inp	Topology sensitivities for C3D8 elements with relative material density as design variables.

Adjoint design sensitivity analysis for implicit transient dynamic procedures

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

CPS4 C3D8 S4

Features tested

This section includes a general set of simple tests to verify the adjoint design sensitivity analysis techniques for shell and solid continuum elements for an implicit transient dynamic step. Geometric nonlinearity, contact (with friction), and multi-point constraints are used for the models. A full range of design variables are used, including those related to topology and sizing sensitivities.

Problem description

All tests use elastic constitutive material properties. The design parameters are either relative material density or shell element thickness. Design responses are defined at a number of time points.

Results and discussion

All adjoint sensitivity results are verified by comparing them to results obtained using finite difference calculations.

Input files

CPS4_dyn_DV_Stiff.inp	Topology sensitivities for CPS4 elements with relative material density as design variables.
S4_dyn_DV_Stiff.inp	Topology sensitivities for S4 elements with relative material density as design variables.
S4_dyn_DV_Thick.inp	Sizing sensitivities for S4 elements with shell thickness as design variables.
C3D8_dyn_MD_DV_Node.inp	Shape sensitivities for C3D8 elements in a moderate dissipation application transient dynamic analysis with nodal positions as design variables.
C3D8_dyn_QS_DV_Node.inp	Shape sensitivities for C3D8 elements in a quasi-static application transient dynamic analysis with nodal positions as design variables.
C3D8_dyn_TF_DV_Node.inp	Shape sensitivities for C3D8 elements in a transient fidelity application transient dynamic analysis with nodal positions as design variables.
CPS4_dyn_Cont_Nonlin_DV_Node.inp	Shape sensitivities for CPS4 elements in a nonlinear transient dynamic analysis involving contact with nodal positions as design variables.
S4_dyn_MPC_DV_Node.inp	Bead sensitivities for S4 elements in a transient dynamic analysis involving multi-point constraints with nodal positions as design variables.

Adjoint design sensitivity analysis for combined Abaqus and fe-safe–based fatigue procedures

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D10 S3

Features tested

This section includes a simple test to verify the adjoint fatigue design responses and corresponding sensitivity analysis techniques for shell and solid continuum elements for a combined Abaqus and fe-safe–based fatigue procedure. Linear perturbation steps with multiple load cases are utilized in the Abaqus analysis. The Abaqus stress results for these linear load cases form the basis of the fatigue calculations in fe-safe, which lead to the fatigue design response and the corresponding partial derivatives (with respect to the components of the stress tensor) . A number of different design responses, including damage design response, are also tested.

Problem description

The model contains 48 continuum elements and 8 shell elements, all with linear elastic material response. It includes two load cases where the shell elements are subjected to pressure loads, which are then both carried over to drive fe-safe fatigue calculations. The design variables are mixed and include the following types:

- shell element thickness (for sizing sensitivities),
- nodal position (for shape/bead sensitivities), and
- relative material density that scales the constitutive material properties and stress relaxation (for topology sensitivities).

Results and discussion

All adjoint sensitivity results are verified by comparing to results using finite difference calculations.

Input files

Fatigue_static_DV_Node_Thick_Stiff.inp	Bead, shape, sizing, and topology for C3D10 and S3 elements for linear load cases with nodal position, shell thickness, and relative material density as design variables. The fatigue damage element responses are computed in fe-safe based on the Abaqus stress results for the two linear load cases. The macro and project files are necessary for the fe-safe fatigue calculations.
Fatigue_static_DV_Node_Thick_Stiff.macro	Fatigue macro driving the damage calculations using fe-safe.

Use the `abaqus fetch` command (see *Fetching Sample Input Files*) to extract the following file:

Fatigue_static_DV_Node_Thick_Stiff.project.7z

Fatigue project for damage calculations using fe-safe.

Transferring results between Abaqus/Standard and Abaqus/Explicit

In this section:

- *Transferring results between Abaqus/Explicit and Abaqus/Standard*
- *Transferring results from one Abaqus/Standard analysis to another Abaqus/Standard analysis*
- *Transferring results from one Abaqus/Explicit analysis to another Abaqus/Explicit analysis*
- *Transferring results with general beam sections*
- *Transferring results with general shell sections*
- *Adding and removing elements during results transfer*
- *Transferring rigid elements*
- *Transferring mass and rotary inertia elements*
- *Transferring connector elements into Abaqus/Explicit*
- *Transferring hourglass forces*
- *Changing the material definition during import*
- *Transferring results with plasticity*
- *Transferring results with damage*
- *Transferring results with hyperelasticity*
- *Transferring results with viscoelasticity*
- *Transferring results for a hyperelastic sheet with a circular hole*
- *Transferring results with hyperfoam*
- *Transferring results with orientation*
- *Miscellaneous results transfer tests*
- *Transferring results with parallel rheological framework*
- *Transferring results with nodal temperature and field variables*
- *Transferring results of an analysis model multiple times*
- *Transferring results from multiple previous analyses*

Transferring results between Abaqus/Explicit and Abaqus/Standard

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Transferring stress/displacement results from Abaqus/Explicit to Abaqus/Standard

Elements tested

B21 B22 B31 B32 C3D4 C3D6 C3D8R C3D8 CAX3 CAX4R
CPE3 CPE4R CPS3 CPS4R M3D3 M3D4 M3D4R
S3R S4R S4 SAX1 T2D2 T3D2
C3D10M CPE6M CPS6M CAX6M SC6R SC8R
COH2D4 COHAX4 COH3D8 COH3D6

Problem description

The verification tests in this section consist of one-element models that are subjected to tensile, pure shear, or bending loads in Abaqus/Explicit. The analyses in Abaqus/Explicit are followed by analyses in Abaqus/Standard in which the results are imported from the Abaqus/Explicit analysis and the loading is removed. Nearly all the tests involve purely elastic materials. The tests are performed for all combinations of parameters to check whether the reference configuration is reset to be the imported configuration, and the material state is to be imported upon importing information from a previous Abaqus analysis. To verify that the results from the Abaqus/Explicit analyses are imported correctly into Abaqus/Standard, the results of the Abaqus/Standard analyses should show that the elements return to their original configuration before the loading in the Abaqus/Explicit analysis, except when the material state is not imported, in which case the elements remain in the deformed configuration.

The sequence of loading in Abaqus/Explicit and unloading in Abaqus/Standard is illustrated in *Figure 1*

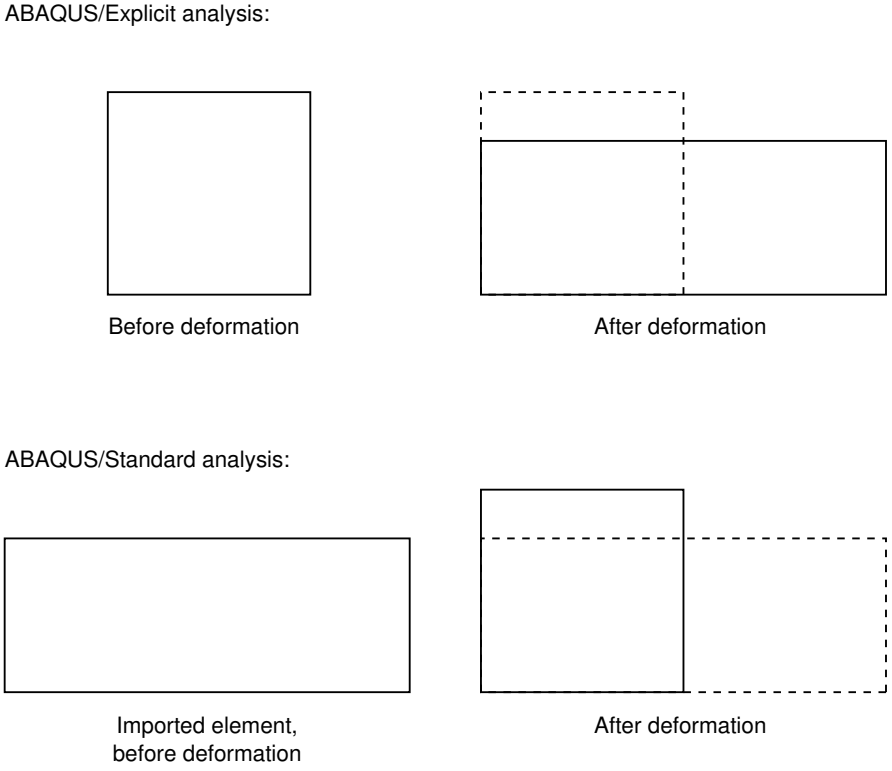


Figure 1: Sequence of loading and unloading.

for an S4R element loaded in tension when the reference configuration is updated to be the imported configuration, and the current material state is imported.

The loading is applied in the Abaqus/Explicit analyses by prescribing the appropriate displacements. In the Abaqus/Standard analyses all the boundary conditions must be redefined, and in all cases only the fixed boundary conditions are defined. The shell and membrane elements are loaded so that the maximum displacements are 2. The remaining elements are loaded so that the maximum displacements are 0.2.

Analyses with reduced-integration elements require hourglass control to remove singular (hourglass) modes. However, differences in the hourglass forces computed in Abaqus/Explicit and Abaqus/Standard affect the force equilibrium for imported problems. Using enhanced hourglass control for both the Abaqus/Explicit and Abaqus/Standard analyses minimizes the differences in the hourglass forces upon import. Verification tests with enhanced hourglass control for both the Abaqus/Explicit and Abaqus/Standard analyses are included to test the performance of import problems.

The material model used for nearly all the tests is isotropic linear elasticity. One test consists of a plastic material modeled with Mises plasticity. The material properties used are as follows:

Young's modulus = 200×10^9

Poisson's ratio = 0.3

Density = 1000.

Yield stress = 500×10^6

Hardening = 500×10^6

For the tests using cohesive elements, some use elasticity with uncoupled traction behavior, some use hyperelasticity, some include damage, and the tests with pure shear loading also use additional transverse shear stiffness.

Results and discussion

In all the elastic tests that involve tensile loading, shear loading, and bending, the elements return to their original configuration. In some cases tighter controls for the convergence criteria are enforced for the Abaqus/Standard analyses to obtain more accurate results.

The elastic tests for shell and membrane elements show differences when comparing the section thickness that is computed by Abaqus/Standard and the original thickness. In Abaqus/Explicit the changes in shell and membrane thickness are computed using the material Poisson's ratio, while in Abaqus/Standard the default is to compute the thickness based on the assumption of no volume change. In most practical cases the thickness change during unloading or springback will be small: the differences observed in these tests occur because the material is assumed to remain elastic for very large deformations.

Input files

The input file names describe the analysis procedure, element type, the load type, and the options specified upon import.

The first two characters indicate that the results are always transferred from Abaqus/Explicit to Abaqus/Standard.

The Abaqus/Explicit analysis files follow the format xs_x_*element_load*.inp; the Abaqus/Standard analysis files follow the format xs_s_*element_load_update_state*.inp, where *element* indicates the element type used in the analysis; *load* indicates a key for the type of loading in the analysis, t for tension or s for pure shear (the SAX1, T2D2, and T3D2 elements are tested only in tension); and *update* and *state* indicate the value of these parameters, y for yes and n for no.

In addition to the tension and pure shear tests, the S4R, B21, and B31 elements are loaded in bending (indicated by a b in the *load* position of the input file name), and the B31 element is also loaded in twist (indicated by a w in the *load* position of the input file name). The use of the plasticity material model (indicated by appending pl to the input file name) is tested for the S4R element loaded in tension and in bending. The use of the OFFSET parameter on the **SHELL SECTION* option (indicated by appending offset to the input file name) is tested for the S4R element loaded in tension only.

Transferring acoustic results from Abaqus/Explicit to Abaqus/Standard

Elements tested

AC2D3 AC2D4R AC3D4 AC3D6 AC3D8R
ACAX3 ACAX4R ACIN2D2 ACIN3D3 ACIN3D4 ACINAX2

Problem description

Compatible solid elements and acoustic elements are tied together. The solid elements are constrained on the face that is opposite to the face tied to the acoustic elements. The acoustic elements are subjected to a pressure loading with a sinusoidal amplitude. After import, the analysis is continued as a dynamic analysis in Abaqus/Standard. Since acoustic elements have no material state and have only pressure degrees of freedom, the pressure values will be imported if the reference configuration is updated to be the imported configuration. If not, they will be set to zero.

Results and discussion

The import analysis is verified by comparing the results from the zero increment of the imported analysis to the last increment of the previous analysis. The results are further verified by continuing the original analysis for a certain period of time after import and checking those results against the imported analysis.

Input files

xs_x_ac3d8_y_n.inp	Pressure load, STATE=YES, UPDATE=NO.
xs_s_ac3d8_y_n.inp	Pressure load, STATE=YES, UPDATE=NO.
xs_x_ac3d4_y_n.inp	Pressure load, STATE=YES, UPDATE=NO.
xs_s_ac3d4_y_n.inp	Pressure load, STATE=YES, UPDATE=NO.
xs_x_ac3d6_y_y.inp	Pressure load, STATE=YES, UPDATE=YES.
xs_s_ac3d6_y_y.inp	Pressure load, STATE=YES, UPDATE=YES.
xs_x_acin3d3_y_n.inp	Pressure load, STATE=YES, UPDATE=NO.
xs_s_acin3d3_y_n.inp	Pressure load, STATE=YES, UPDATE=NO.
xs_x_acin3d4_y_n.inp	Pressure load, STATE=YES, UPDATE=NO.
xs_s_acin3d4_y_n.inp	Pressure load, STATE=YES, UPDATE=NO.
xs_x_ac2d4_y_n.inp	Pressure load, STATE=YES, UPDATE=NO.
xs_s_ac2d4_y_n.inp	Pressure load, STATE=YES, UPDATE=NO.
xs_x_acax4_y_n.inp	Pressure load, STATE=YES, UPDATE=NO.
xs_s_acax4_y_n.inp	Pressure load, STATE=YES, UPDATE=NO.
xs_x_ac2d4_freq.inp	Pressure load, STATE=YES, UPDATE=NO.
xs_s_ac2d4_freq_y_n.inp	Frequency analysis, STATE=YES, UPDATE=NO.

Transferring stress/displacement results from Abaqus/Standard to Abaqus/Explicit

Elements tested

B21 B22 B31 B32 C3D4 C3D6 C3D8 C3D8R CAX3 CAX4R
CPE3 CPE4R CPS3 CPS4R M3D3 M3D4 M3D4R
PIPE21 PIPE31 S3R S4 S4R SAX1 T2D2 T3D2
C3D10M CPE6M CPS6M CAX6M

Problem description

The verification tests in this section are similar to the ones performed in the first section. One-element models are subjected to tensile, pure shear, or bending loads in Abaqus/Standard. The results of these analyses are then imported into Abaqus/Explicit, and the loading is removed. Nearly all the tests involve purely elastic materials. The tests are performed for all combinations of the import capability. To verify that the results from the Abaqus/Standard analyses are imported correctly into Abaqus/Explicit, the results of the Abaqus/Explicit analysis should show that the model oscillates about a mean position when importing the material state. This mean position is the original configuration before the loading in the Abaqus/Standard analysis.

The loading is applied in the Abaqus/Standard analyses by prescribing the appropriate displacements. In the Abaqus/Explicit analyses all the boundary conditions must be redefined, and in all cases only the fixed boundary conditions are defined. All elements are loaded so that the maximum displacements are 0.2.

Verification tests with enhanced hourglass control for both the Abaqus/Explicit and Abaqus/Standard analyses are included to test the performance of import problems.

The material model used is the same as the one used in the previous section.

Results and discussion

In all cases the stresses are found to be continuous across the respective Abaqus/Standard and Abaqus/Explicit analyses when importing the material state. The displacements, strains, and energy quantities such as the recoverable strain energy are verified to be continuous across the two analyses when the reference configuration

is not updated. At the beginning of the Abaqus/Explicit analysis the displacements and strains start from zero when the reference configuration is updated, whereas the stresses are set to zero if the material state is not imported.

Input files

The input file names describe the analysis procedure, the element type, the load type, and the values of the UPDATE and STATE parameters on the **IMPORT* option.

The first two characters indicate that the results are always transferred from Abaqus/Standard to Abaqus/Explicit.

The Abaqus/Standard analysis files follow the format sx_s_*element_load*.inp; the Abaqus/Explicit analysis files follow the format sx_x_*element_load_update_state*.inp, where *element* indicates the element type used in the analysis; *load* indicates a key for the type of loading in the analysis, t for tension or s for pure shear (the SAX1, T2D2, and T3D2 elements are tested only in tension); and *update* and *state* indicate the value of these parameters, y for yes and n for no.

In addition to the tension and pure shear tests, B21, B31, and S4R elements are loaded in bending (indicated by a b in the *load* position of the input file name); and B31 elements are also loaded in twist (indicated by a w in the *load* position of the input file name).

Transferring acoustic results from Abaqus/Standard to Abaqus/Explicit

Elements tested

AC2D3 AC2D4 AC3D4 AC3D6 AC3D8
ACAX3 ACAX4 ACIN2D2 ACIN3D3 ACIN3D4 ACINAX2

Problem description

Compatible solid elements and acoustic elements are tied together. The solid elements are constrained on the face that is opposite to the face tied to the acoustic elements. The acoustic elements are subjected to a pressure loading with a sinusoidal amplitude. After import, the analysis is continued as a dynamic analysis in Abaqus/Standard. Since acoustic elements have no material state and have only pressure degrees of freedom, importing the material state will have no effect, while updating the reference configuration to be the imported configuration will import the pressure values. If the reference configuration is not updated, the pressure values will be set to zero.

Results and discussion

The import analysis is verified by comparing the results from the zero increment of the imported analysis to the last increment of the previous analysis. The results are further verified by continuing the original analysis for a certain period of time after import and checking those results against the imported analysis.

Input files

sx_s_ac3d8_y_n.inp	Pressure load, STATE=YES, UPDATE=NO.
sx_x_ac3d8_y_n.inp	Pressure load, STATE=YES, UPDATE=NO.
sx_s_ac3d4_y_n.inp	Pressure load, STATE=YES, UPDATE=NO.
sx_x_ac3d4_y_n.inp	Pressure load, STATE=YES, UPDATE=NO.
sx_s_ac3d6_y_y.inp	Pressure load, STATE=YES, UPDATE=YES.
sx_x_ac3d6_y_y.inp	Pressure load, STATE=YES, UPDATE=YES.
sx_s_acin3d3_y_n.inp	Pressure load, STATE=YES, UPDATE=NO.
sx_x_acin3d3_y_n.inp	Pressure load, STATE=YES, UPDATE=NO.
sx_s_acin3d4_y_n.inp	Pressure load, STATE=YES, UPDATE=NO.

sx_x_acin3d4_y_n.inp	Pressure load, STATE=YES, UPDATE=NO.
sx_s_ac2d4_y_n.inp	Pressure load, STATE=YES, UPDATE=NO.
sx_x_ac2d4_y_n.inp	Pressure load, STATE=YES, UPDATE=NO.
sx_s_acax4_y_n.inp	Pressure load, STATE=YES, UPDATE=NO.
sx_x_acax4_y_n.inp	Pressure load, STATE=YES, UPDATE=NO.

Transferring results from one Abaqus/Standard analysis to another Abaqus/Standard analysis

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Transferring results between *STATIC procedures

Elements tested

C3D4 C3D4H C3D5 C3D5H C3D6 C3D6H C3D8 C3D8H C3D8I C3D8IH C3D10 C3D10H C3D10HS
C3D10M C3D10MH C3D15 C3D15H C3D20 C3D20H C3D27 C3D27H
CAX3 CAX3H CAX4 CAX4H CAX4I CAX4IH CAX6 CAX6H CAX6M CAX6MH CAX8 CAX8H
CPE3 CPE3H CPE4 CPE4H CPE4I CPE4IH CPE6 CPE6H CPE6M CPE6MH CPE8 CPE8H

CPS3 CPS4 CPS4I CPS6 CPS6M CPS8
M3D3 M3D4R S3R S4R SAX1 T2D2 T3D2 SC6R SC8R
COH2D4 COHAX4 COH3D6 COH3D8
CSS8

Problem description

The verification tests outlined in this section are carried out for each element type listed. The finite element model consists of two elements subjected to increasing tensile loads. The first analysis consists of two static steps. During the first step one element is subjected to half the load that is applied to the other element. During the second step both elements are subjected to the same final loads. The results from the end of the first step of this analysis are transferred to a second analysis where the same loads as in the second step of the first analysis are applied to the two elements. The tests are performed for all combinations of resetting the reference configuration to be the imported configuration option and importing the material state option of the import feature. The results at the end of the second analysis should be identical to the results at the end of the first analysis when the material state is imported and the reference configuration is not updated to be the imported configuration.

Verification tests using first-order reduced-integration elements with enhanced hourglass control for both Abaqus/Standard analyses are included to test the import of hourglass forces.

For all the tests other than the ones using cohesive elements with traction separation, the material model uses isotropic linear elasticity, together with Mises plasticity. The material properties used are as follows:

Young's modulus = 200×10^9

Poisson's ratio = 0.3

Density = 7800.

Yield stress = 300×10^6

For the tests using cohesive elements with traction separation, the material model uses linear elasticity with uncoupled traction, together with a damage model.

Results and discussion

The results from the import analysis with the material state imported and the reference configuration not updated are identical to the results from the end of the first analysis. In all cases when the material state is imported, the stresses, elastic strains, and equivalent plastic strains are continuous during the transfer from the first analysis to the second analysis. The displacements, strains, and energy quantities such as the recoverable strain energy are continuous across the two analyses when the reference configuration is not updated. At the beginning of the second Abaqus/Standard analysis the displacements and strains start from zero if the reference configuration is updated to be the imported configuration; the elastic strains, stresses, and equivalent plastic strains are set to zero if the material state is not imported.

Input files

The input file names describe the analysis procedure, the element type, and the values of the options specified upon import.

The first two characters indicate that the results are always transferred from one Abaqus/Standard analysis to another Abaqus/Standard analysis. The third character, which is a number, indicates the analysis stage: 1 for the original analysis and 2 for the first import analysis.

The first Abaqus/Standard analysis files follow the format ss1_*element*.inp; the second Abaqus/Standard analysis files follow the format ss2_*element_update_state*.inp, where *element* indicates the element type used in the analysis; and *update* and *state* indicate the value of these parameters, y for yes and n for no.

Transferring results between direct-integration implicit dynamic procedures

Elements tested

C3D4 C3D5 C3D6 C3D8 C3D10 C3D10HS CAX3 CAX4 CAX6 CPE3 CPE4 CPE6 CPS3 CPS4 CPS6
M3D3 M3D4R S3R S4R SAX1 T2D2 T3D2 SC8R
COH2D4 COHAX4 COH3D6 COH3D8

Problem description

The verification tests outlined in this section are carried out for each element type listed. The finite element model consists of a single element subjected to increasing loads. During the first analysis the element is subjected to loads over two direct-integration implicit dynamic steps. The results from the end of the first step of this analysis are transferred to a second analysis in which the element is subjected to the same load prescribed in the second step of the first analysis. The tests are performed for all combinations of resetting the reference configuration to be the imported configuration option and importing the material state option of the import feature. The results at the end of the second analysis should be identical to the results at the end of the first analysis when the material state is imported and the reference configuration is not updated.

The material model used for all the tests is isotropic linear elasticity, together with Mises plasticity. The material properties used are as follows:

Young's modulus = 200×10^9

Poisson's ratio = 0.3

Density = 7800.

Yield stress = 300×10^6

Results and discussion

The results from the import analysis with the material state imported and the reference configuration not updated are identical to the results from the end of the first Abaqus/Standard analysis. In all cases when the material state is imported, the stresses, elastic strains, and equivalent plastic strains are continuous during the transfer from the first analysis to the second analysis. The displacements, velocities, strains, and energy quantities such as the recoverable strain energy are continuous across the two analyses when the reference configuration is not updated. At the beginning of the second analysis the displacements and strains start from zero if the reference configuration is updated; the stresses, elastic strains, and equivalent plastic strains are set to zero if the material state is not imported.

Input files

The input file names describe the analysis procedure, the element type, the load type, and the values of the options specified upon import.

The first two characters indicate that the results are always transferred from one Abaqus/Standard analysis to another Abaqus/Standard analysis.

The first Abaqus/Standard analysis files follow the format ss1_*element*_dd_*load*.inp; the second Abaqus/Standard analysis files follow the format ss2_*element*_dd_*load_update_state*.inp, where *element* indicates the element type used in the analysis; *load* indicates a key for the type of loading in the analysis if it is other than tension (s for pure shear, w for twist; the *load* key is omitted for tension loading); and *update* and *state* indicate the value of these parameters, y for yes and n for no.

Transferring results from a direct-integration implicit dynamic procedure to a static procedure

Elements tested

C3D4 C3D5 C3D6 C3D8 C3D10 C3D10HS CAX3 CAX4 CAX6 CPE3 CPE4 CPE6 CPS3 CPS4 CPS6
M3D3 M3D4R S3R S4R SAX1 T2D2 T3D2

Problem description

The verification tests outlined in this section are carried out for each element type listed. The finite element model consists of a single element subjected to increasing loads. During the first analysis the element is subjected to loads over three steps. The first step is a direct-integration implicit dynamic procedure, the second is a static springback step, and the final is a static loading step. The results from the end of the first step of this analysis are transferred to a second analysis in which the element undergoes springback and the final static loading step. The tests are performed using the current material state import option with and without the reference configuration update option of import feature. The results at the end of the import analysis should be identical to the results at the end of the first analysis when the reference configuration is not updated to be the import configuration and the current material state is imported.

The material model used for all the tests is isotropic linear elasticity, together with Mises plasticity. The material properties used are as follows:

Young's modulus = 200×10^9

Poisson's ratio = 0.3

Density = 7800.

Yield stress = 300×10^6

Results and discussion

In all cases the stresses, elastic strains, and equivalent plastic strains at the end of the two analyses are identical when the current material state is imported. The displacements, strains, and energy quantities such as the recoverable strain energy are also identical at the end of the two analyses when the reference configuration is not updated.

Input files

The input file names describe the analysis procedure, the element type, the load type, and the values of the options specified upon import.

The first two characters indicate that the results are always transferred from one Abaqus/Standard analysis to another Abaqus/Standard analysis.

The first Abaqus/Standard analysis files follow the format ss1_*element*_ds_*load*.inp; the second Abaqus/Standard analysis files follow the format ss2_*element* _ds_*load_update_state*.inp, where *element* indicates the element type used in the analysis; *load* indicates a key for the type of loading in the analysis if it is other than tension (s for pure shear, w for twist; the *load* key is omitted for tension loading); and *update* and *state* indicate the value of these parameters, y for yes and n for no.

Transferring temperatures from a fully coupled thermal-stress analysis

Elements tested

C3D4T C3D6T C3D8HT C3D8RT C3D8T CAX3T CAX4HT CAX4RHT CAX4RT CAX4T CAX6MHT CAX6MT CAX8HT CAX8RHT CAX8RT CAX8T CPE3T CPE4HT CPE4RHT CPE4RT CPE4T CPE6MHT CPE6MT CPE8HT CPE8RHT CPE8RT CPE8T CPS3T CPS4RT CPS4T CPS6MT CPS8RT CPS8T S3RT S4RT S4T SC6RT SC8RT

Problem description

The verification tests outlined in this section are carried out for each element type listed. The finite element model consists of a single element subjected to thermal loads. The first analysis has prescribed initial conditions for the temperature of the element. The element is subjected to a combination of concentrated loads and temperatures during a fully coupled thermal stress step. The results from the end of this analysis are transferred to a second analysis that consists of two fully coupled thermal stress steps. In this analysis a new element is defined with the same material properties and initial conditions that the imported element had at the beginning of the first analysis. In the first step of the import analysis all the degrees of freedom of the imported element are held fixed and the new element is subjected to the same loads as in the first analysis. During the second step of the import analysis both elements are subjected to identical loads. The tests are performed with the material state imported and the reference configuration both updated and not updated.

The material model used for all the tests is isotropic linear elasticity, together with Mises plasticity. The thermal properties of the material are also taken to be isotropic.

Results and discussion

The results for both elements at the end of the second analysis are identical when the material state is imported and the reference configuration is not updated. When the material state is imported and the reference configuration is updated, the results for the two elements are identical for the stresses, elastic strains, and equivalent plastic strains; the thermal strains and total strains differ due to the updated reference configuration.

Input files

The input file names describe the analysis procedure, the element type, the load type, and the values of the options specified upon import.

The first two characters indicate that the results are always transferred from one Abaqus/Standard analysis to another Abaqus/Standard analysis.

The first Abaqus/Standard analysis files follow the format ss1_*element*_ctd.inp; the second Abaqus/Standard analysis files follow the format ss2_*element*_ctd_*update_state*.inp, where *element* indicates the element type used in the analysis and *update* and *state* indicate the value of these parameters, y for yes and n for no.

Transfer of predefined fields

Elements tested

C3D8 C3D8R C3D10 C3D10HS CAX4 CPE4 CPE4R CPS4 CPS4R M3D4R S4R

Problem description

The verification tests outlined in this section are carried out for each element type listed. During the first analysis a single element has prescribed initial conditions for temperature and a field variable. The material associated with the element has temperature- and field-variable-dependent properties. The element develops stresses when the temperature and the field variable values are changed over the analysis step. The results from the end of this analysis are transferred to a second analysis. In the second analysis a new element is defined with the same material properties and initial conditions that the imported element had at the beginning of the first analysis. During the import analysis both elements are subjected to the same final values for the temperature and field variable. The tests are performed with the material state imported and the reference configuration both updated and not updated.

The material model used for all the tests is isotropic linear elasticity, together with Mises plasticity. Both properties depend on the prescribed temperature and a field variable.

Results and discussion

The results for both elements at the end of the second analysis are identical when the material state is imported and the reference configuration is not updated. When the material state is imported and the reference configuration is updated, the results for the two elements are identical for the stresses, elastic strains, and equivalent plastic strains; the thermal strains and total strains differ due to the updated reference configuration.

Input files

The input file names describe the analysis procedure, the element type, and the import settings.

The first two characters indicate that the results are always transferred from one Abaqus/Standard analysis to another Abaqus/Standard analysis.

The first Abaqus/Standard analysis files follow the format ss1_*element*_tfv.inp; the second Abaqus/Standard analysis files follow the format ss2_*element*_tfv_*update_state*.inp, where *element* indicates the element type used in the analysis and *update* and *state* indicate the value of these parameters, y for yes and n for no.

Transfer of contact conditions

Elements tested

C3D8R C3D10 C3D10HS CAX4R CPE4R CPS4R S4R

Problem description

The verification tests in this section consist of two analyses for each element type listed. The first analysis consists of two blocks of elements initially separated by a small distance. During the first step of the analysis the two blocks are brought together to establish contact. During the second step the blocks are made to slide relative to each other. The material associated with the element is elastic-plastic; the interface between the two blocks has a coefficient of friction of 0.1. The results from the end of the first step of this analysis are transferred to a second analysis in which the two blocks are made to slide relative to each other in a manner identical to that in the second step of the first analysis. The tests are performed with the material state imported and the reference configuration both updated and not updated.

The material model used for all the tests is isotropic linear elasticity, together with Mises plasticity.

Results and discussion

The results at the end of the import analysis with the material state imported and the reference configuration not updated are identical to the results at the end of the original analysis. When the material state is imported and the reference configuration is updated, the results for the two analyses are identical for the contact stresses; the values for the relative slip of the surfaces differ due to the updated reference configuration.

Input files

The input file names describe the element type and the import settings.

The first two characters indicate that the results are always transferred from one Abaqus/Standard analysis to another Abaqus/Standard analysis.

The first Abaqus/Standard analysis files follow the format ss1_*element*_contact.inp or ss1_*element*_contact_auglagr.inp; the second Abaqus/Standard analysis files follow the format ss2_*element*_contact_*update_state*.inp or ss2_*element*_contact_auglagr_*update_state*.inp, where *element* indicates the element type used in the analysis and *update* and *state* indicate the value of these parameters, y for yes and n for no.

Transfer of rebar layers and embedded elements

Elements tested

C3D8 CAX4 CCL12
SAX1 S3R S4R
M3D3 M3D4R
SFMAX1 SFM3D3 SFM3D4R SFMCL6

Problem description

The tests outlined in this section verify the accuracy of the transfer of rebar layers and embedded elements from one Abaqus/Standard analysis to another Abaqus/Standard analysis. The tests are performed for each of the elements listed above.

The tests involve elements with rebar layers or embedded elements subjected to loading over two static steps in the first analysis. The results from the end of the first step are then transferred to another Abaqus/Standard static import analysis. In addition to the imported elements, new elements with rebar layers or embedded elements are defined in the import analysis. These new elements are identical to the initial element definitions of the imported elements in the original analysis. During the import analysis, the imported and the newly defined elements are subjected to loads such that the final loads are identical to those applied at the end of the second step in the original analysis. The import analysis is performed with the material state imported and the reference configuration both updated and not updated, except for cylindrical elements for which the reference configuration is not updated.

Results and discussion

The results for the two sets of elements in the import analysis—that is, the newly defined elements and the imported elements—are identical at the end of the analysis when the material state is imported and the reference configuration is not updated. In addition, these results are identical to the results at the end of the second step of the original analysis. These tests demonstrate that appropriate quantities in the rebar layer and embedded elements—such as the stresses, rebar orientations, strains, etc.—are transferred accurately from one Abaqus/Standard analysis to another. The only difference in the results at the end of the import analysis when the reference configuration is updated compared to when it is not updated are in the kinematic quantities such as the total strains, rebar rotations, etc. When the reference configuration is updated in the import analysis, the reference configuration is updated so that the total strains and the rebar rotations at the beginning of the import analysis are set to zero; when the reference configuration is not updated, the total strains and the rebar rotations are continuous across the transfer from one analysis code to another.

Input files

ss1_rebar_memb.inp	First Abaqus/Standard analysis.
ss2_rebar_memb_n_y.inp	Abaqus/Standard import analysis with UPDATE=NO and STATE=YES.
ss2_rebar_memb_y_y.inp	Abaqus/Standard import analysis with UPDATE=YES and STATE=YES.
ss1_rebar_memb_embed.inp	First Abaqus/Standard analysis.
ss2_rebar_memb_embed_n_y.inp	Abaqus/Standard import analysis with UPDATE=NO and STATE=YES.
ss2_rebar_memb_embed_y_y.inp	Abaqus/Standard import analysis with UPDATE=YES and STATE=YES.
ss1_rebar_shell.inp	First Abaqus/Standard analysis.
ss2_rebar_shell_n_y.inp	Abaqus/Standard import analysis with UPDATE=NO and STATE=YES.
ss2_rebar_shell_y_y.inp	Abaqus/Standard import analysis with UPDATE=YES and STATE=YES.
ss1_rebar_shellax.inp	First Abaqus/Standard analysis.
ss2_rebar_shellax_n_y.inp	Abaqus/Standard import analysis with UPDATE=NO and STATE=YES.
ss2_rebar_shellax_y_y.inp	Abaqus/Standard import analysis with UPDATE=YES and STATE=YES.
ss1_rebar_surf.inp	First Abaqus/Standard analysis.
ss2_rebar_surf_n_y.inp	Abaqus/Standard import analysis with UPDATE=NO and STATE=YES.
ss2_rebar_surf_y_y.inp	Abaqus/Standard import analysis with UPDATE=YES and STATE=YES.
ss1_rebar_surfax.inp	First Abaqus/Standard analysis.
ss2_rebar_surfax_n_y.inp	Abaqus/Standard import analysis with UPDATE=NO and STATE=YES.
ss2_rebar_surfax_y_y.inp	Abaqus/Standard import analysis with UPDATE=YES and STATE=YES.

ss1_rebar_surf_embed.inp	First Abaqus/Standard analysis.
ss2_rebar_surf_embed_n_y.inp	Abaqus/Standard import analysis with UPDATE=NO and STATE=YES.
ss2_rebar_surf_embed_y_y.inp	Abaqus/Standard import analysis with UPDATE=YES and STATE=YES.
ss1_rebar_ccl12.inp	First Abaqus/Standard analysis.
ss2_rebar_ccl12_n_y.inp	Abaqus/Standard import analysis with UPDATE=NO and STATE=YES.

Transferring results from one Abaqus/Explicit analysis to another Abaqus/Explicit analysis

Products: Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Transferring results between explicit dynamic procedures with nonlinear geometry

Elements tested

B21 B22 B31 B32 C3D4 C3D4H C3D5 C3D6 C3D8 C3D8I C3D8R C3D10M
CAX3 CAX4R CAX6M CPE3 CPE4R CPE6M CPS3 CPS4R CPS6M
M3D3 M3D4R M3D4 S3R S3RS S4 S4R S4RS S4RSW
SAX1 SC6R SC8R T2D2 T3D2

Problem description

The verification tests outlined in this section are carried out for all element types listed. The finite element model consists of elements subjected to increasing tensile loads. The first analysis consists of a single explicit dynamic step. The results from the end of this step of the analysis are transferred to a second analysis, where further tensile loading is applied. The tests are performed for all combinations of the import capability. The results at the end of the second analysis should be identical to the results at the end of the first analysis when the material state is imported and the reference configuration is not updated. Elements are modeled with a variety of different constitutive models, including isotropic elasticity; anisotropic elasticity; lamina elasticity; orthotropic elasticity; orthotropic elasticity with engineering constants; hyperelasticity with Marlow, Arruda-Boyce, and polynomial potentials; hyperfoams; and equation of state. Hyperelastic models are used in combination with viscoelasticity and Mullins effect considerations. Modeling of inelastic effects includes plasticity and damage with several different initial and evolution criteria.

Results and discussion

The results from the import analysis in which the material state is imported and the reference configuration is not updated are identical to the results from the end of the first analysis. In all cases when the reference configuration is updated, the stresses, elastic strains, and equivalent plastic strains are continuous during the transfer from the first analysis to the second analysis. The displacements, strains, and energy quantities such as the recoverable strain energy are continuous across the two analyses when the reference configuration is not updated. At the beginning of the second Abaqus/Explicit analysis, strains start from zero if the reference configuration is updated; the elastic strains, stresses, and equivalent plastic strains are set to zero if the material state is not imported.

Input files

The input file names describe the analysis procedure, the material type modeled, and the values of the options specified on import.

The first two characters indicate that the results are always transferred from one Abaqus/Explicit analysis to another Abaqus/Explicit analysis. The third character, which is a number, indicates the analysis stage: 1 for the original analysis, and 2 for the first import analysis.

The first Abaqus/Explicit analysis files follow the format xx1_*material*.inp; the second Abaqus/Explicit analysis files follow the format xx2_*material_update_state*.inp, where *material* indicates the material type used in the analysis and *update* and *state* indicate the value of these parameters: y for yes and n for no.

First Abaqus/Explicit analysis files

xx1_elastic.inp	Elements with elastic materials loaded in tension.
xx1_hyper.inp	Elements with hyperelastic materials loaded in tension.
xx1_inelastic.inp	Elements with inelastic materials loaded in tension.
xx_elastic_ef1.inp	Include file with nodal coordinates and set definitions.

Second Abaqus/Explicit analysis files

Elastic materials tests:

xx2_elastic_n_n.inp	Model loaded in tension, UPDATE=NO, STATE=NO.
xx2_elastic_n_y.inp	Model loaded in tension, UPDATE=NO, STATE=YES.
xx2_elastic_y_n.inp	Model loaded in tension, UPDATE=YES, STATE=NO.
xx2_elastic_y_y.inp	Model loaded in tension, UPDATE=YES, STATE=YES.

Hyperelastic materials tests:

xx2_hyper_n_n.inp	Model loaded in tension, UPDATE=NO, STATE=NO.
xx2_hyper_n_y.inp	Model loaded in tension, UPDATE=NO, STATE=YES.
xx2_hyper_y_n.inp	Model loaded in tension, UPDATE=YES, STATE=NO.
xx2_hyper_y_y.inp	Model loaded in tension, UPDATE=YES, STATE=YES.

Inelastic materials tests:

xx2_inelastic_n_n.inp	Model loaded in tension, UPDATE=NO, STATE=NO.
xx2_inelastic_n_y.inp	Model loaded in tension, UPDATE=NO, STATE=YES.
xx2_inelastic_y_n.inp	Model loaded in tension, UPDATE=YES, STATE=NO.
xx2_inelastic_y_y.inp	Model loaded in tension, UPDATE=YES, STATE=YES.

Transferring results between explicit dynamic procedures with linear geometry

Elements tested

B21 B22 B31 B32 C3D4 C3D4H C3D5 C3D6 C3D8 C3D8I C3D8R C3D10M
CAX3 CAX4R CAX6M CPE3 CPE4R CPE6M CPS3 CPS4R CPS6M
M3D3 M3D4R M3D4 S3R S3RS S4 S4R S4RS S4RSW
SAX1 SC6R SC8R T2D2 T3D2

Problem description

The verification tests outlined in this section are carried out for all element types listed. The finite element model consists of elements subjected to increasing tensile loads. The first analysis consists of a single explicit dynamic step. The results from the end of this step of the analysis are transferred to a second analysis, where further tensile loading is applied. The tests are performed for both material state settings of the import capability. The results at the end of the second analysis should be identical to the results at the end of the first analysis when the material state is imported. Elements are modeled with a variety of different constitutive models, including isotropic elasticity, anisotropic elasticity, lamina elasticity, orthotropic elasticity, and orthotropic elasticity with engineering constants.

Results and discussion

The results from the import analysis with the material state imported are identical to the results from the end of the first analysis. In all cases when the material state is imported, the stresses are continuous during the transfer from the first analysis to the second analysis. The displacements strains and energy quantities are continuous across the two analyses. At the beginning of the second Abaqus/Explicit analysis, stresses are set to zero if the material state is not imported.

Input files

First Abaqus/Explicit analysis files

xx1_elastic_lingeom.inp	Elements with elastic materials loaded in tension.
xx_elastic_ef1.inp	Include file with nodal coordinates and set definitions.

Second Abaqus/Explicit analysis files

xx2_elastic_lingeom_n_n.inp	Model loaded in tension, UPDATE=NO, STATE=NO.
xx2_elastic_lingeom_n_y.inp	Model loaded in tension, UPDATE=NO, STATE=YES.

Transferring temperatures from an explicit dynamic procedure

Elements tested

C3D4T C3D6T C3D8RT C3D8T C3D10MT
CAX3T CAX4RT CAX6MT CPE3T CPE4RT CPE6MT CPS3T CPS4RT CPS6MT
SC6RT SC8RT

Problem description

The verification tests outlined in this section are carried out for all element types listed. The finite element model consists of elements subjected to tensile and thermal loads. The first analysis consists of a single fully coupled thermal-stress step. The results from the end of this step of the analysis are transferred to a second analysis, where further tensile loading is applied. The tests are performed with the material state imported and the reference configuration updated and not updated. The results at the end of the second analysis should be identical to the results at the end of the first analysis when the material state is imported and the referenced configuration is not updated. Elements are modeled with a variety of different constitutive models, including isotropic elasticity, anisotropic elasticity, lamina elasticity, orthotropic elasticity, and orthotropic elasticity with engineering constants. The thermal properties of the material are taken to be isotropic.

Results and discussion

Results at the end of the second analysis are identical when the material state is imported and the referenced configuration is not updated. When the material state is imported and the reference configuration is updated, the results are identical for the stresses; the thermal strains and total strains differ due to the updated reference configuration.

Input files

First Abaqus/Explicit analysis files

xx1_tempdisp.inp	Elements subjected to tensile and thermal loads.
xx_elastic_ef1.inp	Include file with nodal coordinates and set definitions.

Second Abaqus/Explicit analysis files

xx2_tempdisp_n_n.inp	Model subjected to tensile and thermal loads, UPDATE=NO, STATE=NO.
xx2_tempdisp_n_y.inp	Model subjected to tensile and thermal loads, UPDATE=NO, STATE=YES.
xx2_tempdisp_y_n.inp	Model subjected to tensile and thermal loads, UPDATE=YES, STATE=NO.
xx2_tempdisp_y_y.inp	Model subjected to tensile and thermal loads, UPDATE=YES, STATE=YES.

Transferring acoustic results

Elements tested

AC2D3 AC2D4R AC3D4 AC3D6 AC3D8R
ACAX3 ACAX4R

Problem description

The acoustic elements are subjected to a linearly increasing pressure loading. Since acoustic elements have no material state, importing the material state has no effect. Acoustic elements have pressure degrees of freedom only; thus, if the reference configuration is updated, the pressure values will be imported. If not, they will be set to zero.

Results and discussion

The import analysis is verified by comparing the results from the zero increment of the imported analysis to the last increment of the previous analysis.

Input files

First Abaqus/Explicit analysis files

xx1_acoustic.inp	Elements subjected to acoustic loads.
xx_elastic_ef1.inp	Include file with nodal coordinates and set definitions.

Second Abaqus/Explicit analysis files

Elastic materials tests:

xx2_acoustic_n_n.inp	Model subjected to acoustic loads, UPDATE=NO, STATE=NO.
xx2_acoustic_n_y.inp	Model subjected to acoustic loads, UPDATE=NO, STATE=YES.
xx2_acoustic_y_n.inp	Model subjected to acoustic loads, UPDATE=YES, STATE=NO.
xx2_acoustic_y_y.inp	Model subjected to acoustic loads, UPDATE=YES, STATE=YES.

Transferring contact conditions

Elements tested

C3D8R C3D10M S4R

Problem description

The verification tests in this section consist of analyses involving contact with analytical rigid surfaces, surface contact, and edge contact. The results from the end of the first step of the analyses are transferred to a second analysis. The tests are performed with the material state imported and the reference configuration updated and not updated.

The material model used for all the tests is isotropic linear elasticity, together with Mises plasticity.

Results and discussion

The results at the beginning of the import analysis with the material state imported and the reference configuration not updated are identical to the results at the end of the original analysis. When the material state is imported and the reference configuration is updated, the results for the two analyses are identical for the contact stresses; the values for the relative slip of the surfaces differ due to the updated reference configuration.

Input files

First Abaqus/Explicit analysis files

xx1_anls.inp	Contact with analytical rigid surface.
xx1_edge.inp	Edge contact.
xx1_facet.inp	Surface contact.

Second Abaqus/Explicit analysis files

Contact with analytical rigid surface tests:

xx2_anls_n_n.inp	Contact with analytical rigid surface, UPDATE=NO, STATE=NO.
xx2_anls_n_y.inp	Contact with analytical rigid surface, UPDATE=NO, STATE=YES.
xx2_anls_y_n.inp	Contact with analytical rigid surface, UPDATE=YES, STATE=NO.
xx2_anls_y_y.inp	Contact with analytical rigid surface, UPDATE=YES, STATE=YES.

Edge contact tests:

xx2_edge_n_n.inp	Edge contact, UPDATE=NO, STATE=NO.
xx2_edge_n_y.inp	Edge contact, UPDATE=NO, STATE=YES.
xx2_edge_y_n.inp	Edge contact, UPDATE=YES, STATE=NO.
xx2_edge_y_y.inp	Edge contact, UPDATE=YES, STATE=YES.

Surface contact tests:

xx2_facet_n_n.inp	Surface contact, UPDATE=NO, STATE=NO.
xx2_facet_n_y.inp	Surface contact, UPDATE=NO, STATE=YES.
xx2_facet_y_n.inp	Surface contact, UPDATE=YES, STATE=NO.
xx2_facet_y_y.inp	Surface contact, UPDATE=YES, STATE=YES.

Transferring rebar layers and embedded elements

Elements tested

C3D8 CAX4
S3R S4R SAX1
M3D3 M3D4R

SFM3D3 SFM3D4R

Problem description

The tests outlined in this section verify the accuracy of the transfer of rebar layers and embedded elements from one Abaqus/Explicit analysis to another Abaqus/Explicit analysis. The tests are performed for all element types listed.

The tests involve elements with rebar layers or embedded elements subjected to loading over two explicit dynamic steps in the first analysis. The results from the end of the first step are then transferred to another Abaqus/Explicit dynamic import analysis. In addition to the imported elements, new elements with rebar layers or embedded elements are defined in the import analysis. These new elements are identical to the initial element definitions of the imported elements in the original analysis. During the import analysis, the imported elements and the newly defined elements are subjected to loads such that the final loads are identical to those applied at the end of the second step in the original analysis. The import analysis is performed with the material state imported and the reference configuration updated and not updated.

Results and discussion

The results for the two sets of elements in the import analysis—that is, the newly defined elements and the imported elements—are identical at the end of the analysis when the material state is imported and the reference configuration is not updated. In addition, these results are identical to the results at the end of the second step of the original analysis. These tests demonstrate that appropriate quantities in the rebar layer and embedded elements—such as the stresses, rebar orientations, strains, etc.—are transferred accurately from one Abaqus/Explicit analysis to another. The only difference in the results at the end of the import analysis when the reference configuration is updated compared to the results when the reference configuration is not updated is in the kinematic quantities such as the total strains, rebar rotations, etc. When the reference configuration is updated in the import analysis, the total strains and the rebar rotations at the beginning of the import analysis are set to zero; when the reference configuration is not updated, the total strains and the rebar rotations are continuous across the transfer from one analysis to another.

Input files

xx1_rebar_memb.inp	First Abaqus/Explicit analysis.
xx2_rebar_memb_n_y.inp	Abaqus/Explicit import analysis with UPDATE=NO, STATE=YES.
xx2_rebar_memb_y_y.inp	Abaqus/Explicit import analysis with UPDATE=YES, STATE=YES.
xx1_rebar_memb_embed.inp	First Abaqus/Explicit analysis.
xx2_rebar_memb_embed_n_y.inp	Abaqus/Explicit import analysis with UPDATE=NO, STATE=YES.
xx2_rebar_memb_embed_y_y.inp	Abaqus/Explicit import analysis with UPDATE=YES, STATE=YES.
xx1_rebar_shell.inp	First Abaqus/Explicit analysis.
xx2_rebar_shell_n_y.inp	Abaqus/Explicit import analysis with UPDATE=NO, STATE=YES.
xx2_rebar_shell_y_y.inp	Abaqus/Explicit import analysis with UPDATE=YES, STATE=YES.
xx1_rebar_shellax.inp	First Abaqus/Explicit analysis.
xx2_rebar_shellax_n_y.inp	Abaqus/Explicit import analysis with UPDATE=NO, STATE=YES.
xx2_rebar_shellax_y_y.inp	Abaqus/Explicit import analysis with UPDATE=YES, STATE=YES.

xx1_rebar_surf.inp	First Abaqus/Explicit analysis.
xx2_rebar_surf_n_y.inp	Abaqus/Explicit import analysis with UPDATE=NO, STATE=YES.
xx2_rebar_surf_y_y.inp	Abaqus/Explicit import analysis with UPDATE=YES, STATE=YES.

Transferring results with general beam sections

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

B21 B31

Problem description

The use of a general beam section to specify section properties for beam elements is verified in the following tests. A B21 and a B31 element are subjected to combined monotonically increasing loads. The analysis consists of a sequential transfer from Abaqus/Standard to Abaqus/Explicit and back to Abaqus/Standard. Nonlinear inelastic section behavior is defined by setting the nonlinear generalized cross section property and omitting the linear and elastic parameters from the options defining axial, bending, and torsional behavior of a general beam section.

The nonlinear inelastic axial, bending, and torsional behavior is defined by the following data lines:

```
0., 0.
2.E7, 0.001
2.5E7, 0.002
3.0E7, 0.003
```

Results and discussion

The results of this analysis demonstrate that section properties specified with a general beam section definition are transferred correctly between Abaqus/Explicit and Abaqus/Standard.

Input files

sx_s_b21_gsp1.inp	First Abaqus/Standard analysis with the B21 element.
sx_x_b21_gsp.inp	Abaqus/Explicit analysis with the B21 element.
xs_s_b21_gsp2.inp	Second Abaqus/Standard analysis with the B21 element.
sx_s_b31_gsp1.inp	First Abaqus/Standard analysis with the B31 element.
sx_x_b31_gsp.inp	Abaqus/Explicit analysis with the B31 element.
xs_s_b31_gsp2.inp	Second Abaqus/Standard analysis with the B31 element.

Transferring results with general shell sections

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Transferring results between Abaqus/Explicit and Abaqus/Standard

Elements tested

S4R

Problem description

Specifying section properties for shell elements using the general shell section definition is verified in the following tests. An S4R element is subjected to simple shear with monotonically increasing loads. The analysis consists of a sequential transfer between Abaqus/Explicit and Abaqus/Standard and back to Abaqus/Explicit. The material parameter is used to define linear isotropic elastic behavior for a general shell section. The composite parameter is used to define orthotropic elastic properties for a shell composed of different linear elastic layers.

The following linear elastic properties are used (the units are not important):

$E_1 = 200 \times 10^9$

$\nu = 0.3$

Density = 7850.

The orthotropic material properties are

$E_1 = 200 \times 10^9$

$E_2 = 100 \times 10^9$

$E_3 = 100 \times 10^9$

$\nu_{12} = 0.3$

$\nu_{13} = 0.23$

$\nu_{23} = 0.34$

$G_{12} = 76.9 \times 10^9$

$G_{13} = 76.9 \times 10^9$

$G_{23} = 9.0 \times 10^9$

Density = 7850.

Verification tests of the enhanced hourglass control method are also included.

Results and discussion

The results of this analysis demonstrate that section properties specified with a general shell section definition are transferred correctly between Abaqus/Explicit and Abaqus/Standard.

Input files

The material properties are specified:

xs_x_s4r_sgm1.inp	First Abaqus/Explicit analysis.
xs_x_s4r_sgm1_enhg.inp	First Abaqus/Explicit analysis with enhanced hourglass control.
xs_s_s4r_sgm.inp	Abaqus/Standard analysis.
xs_s_s4r_sgm_enhg.inp	Abaqus/Standard analysis with enhanced hourglass control.
sx_x_s4r_sgm2.inp	Second Abaqus/Explicit analysis.
sx_x_s4r_sgm2_enhg.inp	Second Abaqus/Explicit analysis with enhanced hourglass control.

The equivalent section properties are input directly and the section stiffness matrix is based on the linear elastic properties given above:

xs_x_s4r_sgd1.inp	First Abaqus/Explicit analysis.
xs_x_s4r_sgd1_enhg.inp	First Abaqus/Explicit analysis with enhanced hourglass control.
xs_s_s4r_sgd.inp	Abaqus/Standard analysis.
xs_s_s4r_sgd_enhg.inp	Abaqus/Standard analysis with enhanced hourglass control.
sx_x_s4r_sgd2.inp	Second Abaqus/Explicit analysis.
sx_x_s4r_sgd2_enhg.inp	Second Abaqus/Explicit analysis with enhanced hourglass control.

The material properties of the layers are specified:

xs_x_s4r_com1.inp	First Abaqus/Explicit analysis.
xs_x_s4r_com1_enhg.inp	First Abaqus/Explicit analysiswith enhanced hourglass control.
xs_s_s4r_com.inp	Abaqus/Standard analysis.
xs_s_s4r_com_enhg.inp	Abaqus/Standard analysis with enhanced hourglass control.
sx_x_s4r_com2.inp	Second Abaqus/Explicit analysis.
sx_x_s4r_com2_enhg.inp	Second Abaqus/Explicit analysis with enhanced hourglass control.

Transferring results from one Abaqus/Standard analysis to another Abaqus/Standard analysis

Elements tested

S4R

Problem description

Specifying section properties for shell elements using the general shell section definition is verified in the following tests, which involve a sequential transfer from one Abaqus/Standard analysis to another. During the first analysis the element is subjected to simple shear by monotonically increasing loads during a static procedure. The results from the end of the first analysis are transferred to a second analysis. During the second analysis a new element is defined and both elements are subjected to the same final loads in a static step. The material state is imported, and the reference configuration is not updated. The material parameter is used to define linear isotropic elastic behavior for a general shell section. The composite parameter is used to define orthotropic elastic properties for a shell composed of different linear elastic layers.

The following linear elastic properties are used (the units are not important):

$E_1 = 200 \times 10^9$

$\nu = 0.3$

Density = 7850.

The orthotropic material properties are

$E_1 = 200 \times 10^9$

$E_2 = 100 \times 10^9$

$E_3 = 100 \times 10^9$

$\nu_{12} = 0.3$

$\nu_{13} = 0.23$

$\nu_{23} = 0.34$

$G_{12} = 76.9 \times 10^9$

$G_{13} = 76.9 \times 10^9$

$G_{23} = 9.0 \times 10^9$

Density = 7850.

Verification tests of the enhanced hourglass control method are also included.

Results and discussion

The results of this analysis demonstrate that section properties specified using the general shell section definition transferred correctly from one Abaqus/Standard analysis to another.

Input files

The material properties are specified:

ss1_s4r_sgm.inp	First Abaqus/Standard analysis.
ss1_s4r_sgm_enhg.inp	First Abaqus/Standard analysis with enhanced hourglass control.
ss2_s4r_sgm.inp	Second Abaqus/Standard analysis.
ss2_s4r_sgm_enhg.inp	Second Abaqus/Standard analysis with enhanced hourglass control.

The equivalent section properties are input directly and the section stiffness matrix is based on the linear elastic properties given above:

ss1_s4r_sgd.inp	First Abaqus/Standard analysis.
ss1_s4r_sgd_enhg.inp	First Abaqus/Standard analysis with enhanced hourglass control.
ss2_s4r_sgd.inp	Second Abaqus/Standard analysis.
ss2_s4r_sgd_enhg.inp	Second Abaqus/Standard analysis with enhanced hourglass control.

The material properties of the layers are specified:

ss1_s4r_com.inp	First Abaqus/Standard analysis.
ss1_s4r_com_enhg.inp	First Abaqus/Standard analysis with enhanced hourglass control.
ss2_s4r_com.inp	Second Abaqus/Standard analysis.

ss2_s4r_com_enhg.inp	Second Abaqus/Standard analysis with enhanced hourglass control.

Adding and removing elements during results transfer

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Transferring results between Abaqus/Explicit and Abaqus/Standard

Elements tested

C3D8R CPE4R S4R

Problem description

The verification problems outlined in this section test the addition and the removal of elements in a sequential import analysis. The problems also test the application of initial stresses and velocities on imported elements that can be applied only under certain conditions (see *Transferring Results between Abaqus/Explicit and Abaqus/Standard*).

The finite element model in these verification problems consists of two elements that are not connected to each other, as shown in *Figure 1*. The elastic, elastic-plastic, hyperelastic, and hyperelastic foam material models are used.

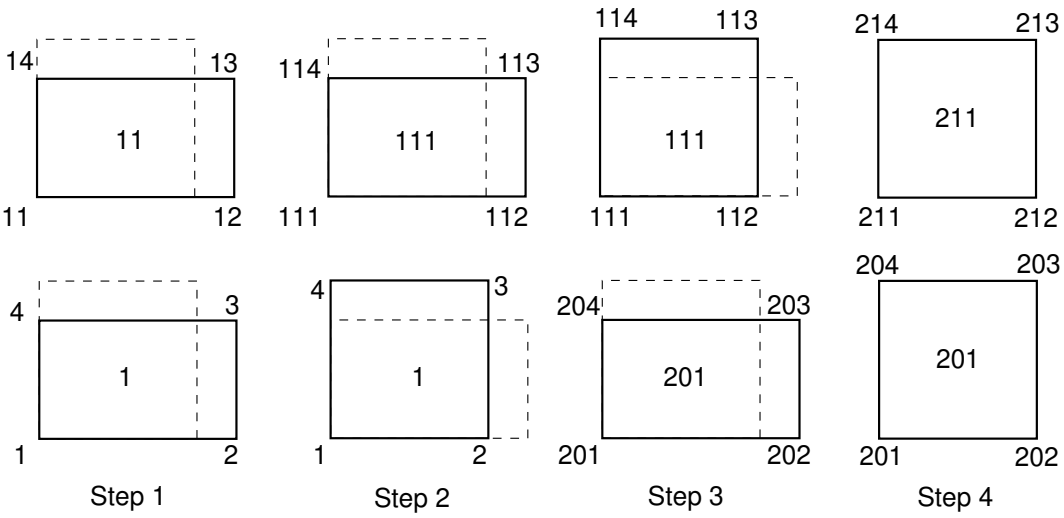


Figure 1: Sequence of loading when adding and removing elements in an import analysis.

Each analysis, for a given combination of an element type and material model, consists of four steps, with the first step being an Abaqus/Explicit analysis. In this step the two elements, 1 and 11, are loaded in tension for all material models except for the hyperelastic foam model, where the elements are loaded in compression.

The second step is an import analysis, with the results being imported into Abaqus/Standard. In this case the results of element 1 only are imported with the reference configuration updated and the material state not imported. Since the material state is not imported, initial stresses can be prescribed for the imported element. In addition, a new element, 111, is defined in the import analysis and subjected to loading in tension (compression when the hyperelastic foam material model is used).

The third step is another import analysis, with the results now being imported into Abaqus/Explicit from the previous Abaqus/Standard analysis. The material state is imported, and the reference configuration is updated. The results for element 111 are imported into the Abaqus/Explicit analysis, while the results for element 1 are not imported. Initial velocities are prescribed at the nodes of the imported element. A new element, 201, is defined in this import analysis and subjected to a tensile load (compressive load when the hyperelastic foam material model is used).

The results of element 201 at the end of the third step are then imported into Abaqus/Standard, with the reference configuration updated and the material state not imported. A new element, 211, is defined in this step. When the reference configuration is updated and the material state is not imported, the nodal definitions and the element connectivities of the imported nodes and elements can be redefined. This feature is tested in the fourth step by modifying the connectivity of element 201 and redefining nodes 203 and 204.

Results and discussion

The tests performed in this section verify that elements can be successfully added and removed in a sequential import analysis.

Input files

C3D8R element tests with an elastic material:

xs_x_c3d8r_ar_el.inp	First Abaqus/Explicit analysis.
xs_s_c3d8r_ar_el.inp	First Abaqus/Standard analysis.
sx_x_c3d8r_ar_el.inp	Second Abaqus/Explicit analysis.
sx_s_c3d8r_ar_el.inp	Second Abaqus/Standard analysis.

CPE4R element tests with an elastic material:

xs_x_cpe4r_ar_el.inp	First Abaqus/Explicit analysis.
xs_s_cpe4r_ar_el.inp	First Abaqus/Standard analysis.
sx_x_cpe4r_ar_el.inp	Second Abaqus/Explicit analysis.
sx_s_cpe4r_ar_el.inp	Second Abaqus/Standard analysis.

S4R element tests with an elastic material:

xs_x_s4r_ar_el.inp	First Abaqus/Explicit analysis.
xs_s_s4r_ar_el.inp	First Abaqus/Standard analysis.
sx_x_s4r_ar_el.inp	Second Abaqus/Explicit analysis.
sx_s_s4r_ar_el.inp	Second Abaqus/Standard analysis.

C3D8R element tests with an elastic-plastic material:

xs_x_c3d8r_ar_ep.inp	First Abaqus/Explicit analysis.
xs_s_c3d8r_ar_ep.inp	First Abaqus/Standard analysis.
sx_x_c3d8r_ar_ep.inp	Second Abaqus/Explicit analysis.
sx_s_c3d8r_ar_ep.inp	Second Abaqus/Standard analysis.

CPE4R element tests with an elastic-plastic material:

xs_x_cpe4r_ar_ep.inp	First Abaqus/Explicit analysis.
xs_s_cpe4r_ar_ep.inp	First Abaqus/Standard analysis.
sx_x_cpe4r_ar_ep.inp	Second Abaqus/Explicit analysis.
sx_s_cpe4r_ar_ep.inp	Second Abaqus/Standard analysis.

S4R element tests with an elastic-plastic material:

xs_x_s4r_ar_ep.inp	First Abaqus/Explicit analysis.
xs_s_s4r_ar_ep.inp	First Abaqus/Standard analysis.
sx_x_s4r_ar_ep.inp	Second Abaqus/Explicit analysis.
sx_s_s4r_ar_ep.inp	Second Abaqus/Standard analysis.

C3D8R element tests with a hyperelastic material:

xs_x_c3d8r_ar_he.inp	First Abaqus/Explicit analysis.
xs_s_c3d8r_ar_he.inp	First Abaqus/Standard analysis.
sx_x_c3d8r_ar_he.inp	Second Abaqus/Explicit analysis.
sx_s_c3d8r_ar_he.inp	Second Abaqus/Standard analysis.

CPE4R element tests with a hyperelastic material:

xs_x_cpe4r_ar_he.inp	First Abaqus/Explicit analysis.
xs_s_cpe4r_ar_he.inp	First Abaqus/Standard analysis.
sx_x_cpe4r_ar_he.inp	Second Abaqus/Explicit analysis.
sx_s_cpe4r_ar_he.inp	Second Abaqus/Standard analysis.

S4R element tests with a hyperelastic material:

xs_x_s4r_ar_he.inp	First Abaqus/Explicit analysis.
xs_s_s4r_ar_he.inp	First Abaqus/Standard analysis.
sx_x_s4r_ar_he.inp	Second Abaqus/Explicit analysis.
sx_s_s4r_ar_he.inp	Second Abaqus/Standard analysis.

C3D8R element tests with hyperelastic foam:

xs_x_c3d8r_ar_hf.inp	First Abaqus/Explicit analysis.
xs_s_c3d8r_ar_hf.inp	First Abaqus/Standard analysis.
sx_x_c3d8r_ar_hf.inp	Second Abaqus/Explicit analysis.
sx_s_c3d8r_ar_hf.inp	Second Abaqus/Standard analysis.

CPE4R element tests with hyperelastic foam:

xs_x_cpe4r_ar_hf.inp	First Abaqus/Explicit analysis.
xs_s_cpe4r_ar_hf.inp	First Abaqus/Standard analysis.
sx_x_cpe4r_ar_hf.inp	Second Abaqus/Explicit analysis.
sx_s_cpe4r_ar_hf.inp	Second Abaqus/Standard analysis.

S4R element tests with hyperelastic foam:

xs_x_s4r_ar_hf.inp	First Abaqus/Explicit analysis.
xs_s_s4r_ar_hf.inp	First Abaqus/Standard analysis.
sx_x_s4r_ar_hf.inp	Second Abaqus/Explicit analysis.
sx_s_s4r_ar_hf.inp	Second Abaqus/Standard analysis.

Transferring results from one Abaqus/Standard analysis to another Abaqus/Standard analysis

Elements tested

C3D8 CPE4 S4R

Problem description

The verification problems outlined in this section test the addition and the removal of elements in a sequential import analysis. The problems also test the application of initial stresses and velocities on imported elements that can be applied only under certain conditions (see *Transferring Results between Abaqus/Explicit and Abaqus/Standard*).

The finite element model in these verification problems consists of two elements that are not connected to each other, as shown in *Figure 1*. Hyperelastic and hyperelastic foam material models are used in the verification problems.

Each analysis, for a given combination of an element type and material model, consists of four steps. In the first step the two elements, 1 and 11, are loaded in tension for the hyperelastic model, while for the hyperelastic foam model the elements are loaded in compression.

The second step is an import analysis, with the results being imported into another Abaqus/Standard static analysis. In this case the results for element 1 only are imported with the updated reference configuration, but the material state is not imported. Since the material state is not imported, initial stresses can be prescribed for the imported element. In addition, a new element, 111, is defined in the import analysis and subjected to loading in tension (compression when the hyperelastic foam material model is used).

The third step is another import analysis, with the results now being imported from the second analysis into an Abaqus/Standard direct-integration implicit dynamic analysis. The material state is imported, and the reference configuration is updated. The results for element 111 are imported into the current analysis, while the results for element 1 are not imported. Initial velocities are prescribed at the nodes of the imported element. A new element, 201, is defined in this import analysis and subjected to a tensile load (a compressive load when the hyperelastic foam material model is used) using the direct-integration implicit dynamic procedure.

The results for element 201 at the end of the third step are then imported into Abaqus/Standard, with the reference configuration updated and the material state not imported. A new element, 211, is defined in this step. When the reference configuration is updated and the material state is not imported, the nodal definitions and the element connectivities for the imported nodes and elements can be redefined. This feature is tested in the fourth step by modifying the connectivity of element 201 and redefining nodes 203 and 204.

The addition and removal of S4R elements with enhanced hourglass control is also tested.

Results and discussion

The tests performed in this section verify that elements can be added and removed successfully in a sequential import analysis.

Input files

C3D8 element tests with a hyperelastic material:

ss1_c3d8_ar_he.inp	First Abaqus/Standard analysis.
ss2_c3d8_ar_he.inp	Second Abaqus/Standard analysis.
ss3_c3d8_ar_he.inp	Third Abaqus/Standard analysis.
ss4_c3d8_ar_he.inp	Fourth Abaqus/Standard analysis.

CPE4 element tests with a hyperelastic material:

ss1_cpe4_ar_he.inp	First Abaqus/Standard analysis.
ss2_cpe4_ar_he.inp	Second Abaqus/Standard analysis.
ss3_cpe4_ar_he.inp	Third Abaqus/Standard analysis.
ss4_cpe4_ar_he.inp	Fourth Abaqus/Standard analysis.

S4R element tests with a hyperelastic material:

ss1_s4r_ar_he.inp	First Abaqus/Standard analysis.
ss1_s4r_ar_he_enhg.inp	First Abaqus/Standard analysis with enhanced hourglass control.
ss2_s4r_ar_he.inp	Second Abaqus/Standard analysis.
ss2_s4r_ar_he_enhg.inp	Second Abaqus/Standard analysis with enhanced hourglass control.
ss3_s4r_ar_he.inp	Third Abaqus/Standard analysis.
ss3_s4r_ar_he_enhg.inp	Third Abaqus/Standard analysis with enhanced hourglass control.
ss4_s4r_ar_he.inp	Fourth Abaqus/Standard analysis.
ss4_s4r_ar_he_enhg.inp	Fourth Abaqus/Standard analysis with enhanced hourglass control.

C3D8 element tests with hyperelastic foam:

ss1_c3d8_ar_hf.inp	First Abaqus/Standard analysis.
ss2_c3d8_ar_hf.inp	Second Abaqus/Standard analysis.
ss3_c3d8_ar_hf.inp	Third Abaqus/Standard analysis.
ss4_c3d8_ar_hf.inp	Fourth Abaqus/Standard analysis.

CPE4 element tests with hyperelastic foam:

ss1_cpe4_ar_hf.inp	First Abaqus/Standard analysis.
ss2_cpe4_ar_hf.inp	Second Abaqus/Standard analysis.
ss3_cpe4_ar_hf.inp	Third Abaqus/Standard analysis.
ss4_cpe4_ar_hf.inp	Fourth Abaqus/Standard analysis.

S4R element tests with hyperelastic foam:

ss1_s4r_ar_hf.inp	First Abaqus/Standard analysis.
ss2_s4r_ar_hf.inp	Second Abaqus/Standard analysis.
ss3_s4r_ar_hf.inp	Third Abaqus/Standard analysis.
ss4_s4r_ar_hf.inp	Fourth Abaqus/Standard analysis.

Transferring rigid elements

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Transferring results between Abaqus/Explicit and Abaqus/Standard

Elements tested

R2D2 R3D3 R3D4

Problem description

The verification problems outlined in this section test the transfer of rigid elements between Abaqus/Explicit and Abaqus/Standard. Contact definitions specified in the original analysis are not imported; they have to be specified again in the import analysis.

These verification problems consist of a deformable block and a rigid surface, as shown in *Figure 1*. The analysis consists of four steps. The first step is performed in Abaqus/Explicit. In this step the block is moved until contact is established between the block and the rigid surface.

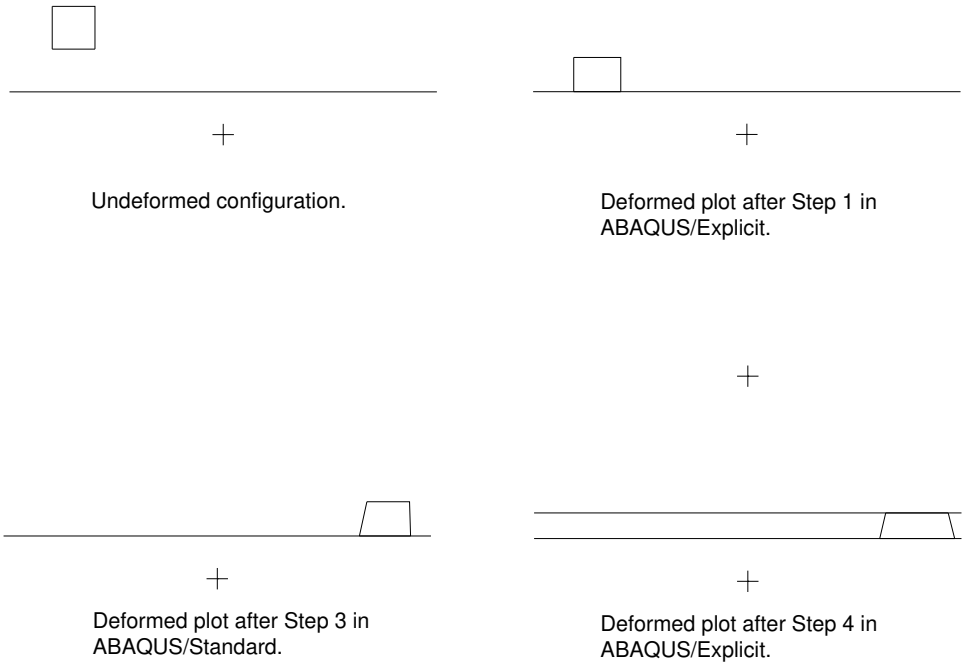


Figure 1: Sequence of loading for the testing of rigid elements.

The results at the end of the Abaqus/Explicit analysis are then imported into Abaqus/Standard with the material state imported and the reference configuration not updated. The contact conditions are redefined since they are not imported. In the second step contact between the block and the rigid surface is resolved. In the third step the block is made to slide on the rigid surface. A coefficient of friction of 0.1 is defined at the contact interface.

The results at the end of the third step of the analysis are then imported into Abaqus/Explicit with the material state imported and the reference configuration not updated. In this step another rigid surface is defined along the top surface of the block. In the course of the Abaqus/Explicit import analysis, the block is compressed between the two rigid surfaces. The sequence of loading is shown in *Figure 1*.

Results and discussion

From these tests it can be seen that rigid elements can be transferred between Abaqus/Explicit and Abaqus/Standard along with their rigid body and reference node definitions. In addition, new rigid elements can be defined in an import analysis.

Input files

R2D2 element tests:

xs_x_r2d2.inp	First Abaqus/Explicit analysis.
xs_s_r2d2.inp	Abaqus/Standard analysis.
sx_x_r2d2.inp	Second Abaqus/Explicit analysis.

R3D3 element tests:

xs_x_r3d3.inp	First Abaqus/Explicit analysis.
xs_x_r3d3_gcont.inp	First Abaqus/Explicit analysis using the general contact capability.
xs_s_r3d3.inp	Abaqus/Standard analysis.
sx_x_r3d3.inp	Second Abaqus/Explicit analysis.
sx_x_r3d3_gcont.inp	Second Abaqus/Explicit analysis using the general contact capability.

R3D4 element tests:

xs_x_r3d4.inp	First Abaqus/Explicit analysis.
xs_x_r3d4_gcont.inp	First Abaqus/Explicit analysis using the general contact capability.
xs_s_r3d4.inp	Abaqus/Standard analysis.
sx_x_r3d4.inp	Second Abaqus/Explicit analysis.
sx_x_r3d4_gcont.inp	Second Abaqus/Explicit analysis using the general contact capability.

Transferring results from one Abaqus/Standard analysis to another Abaqus/Standard analysis

Elements tested

R2D2 R3D3 R3D4

Problem description

The verification problems outlined in this section test the transfer of rigid elements and contact definitions from one Abaqus/Standard analysis to another. The contact definitions and the contact state from the first analysis are transferred to the import analysis. Therefore, the contact definitions do not need to be redefined in the import analysis.

The finite element model consists of a block of deformable material initially located a small distance above a rigid surface. The rigid surface is defined using one of the rigid element types listed. The first step of the first analysis is a static step, in which the deformable block is moved down toward the rigid surface so that contact is established. During the second step of this analysis the block is moved parallel to the rigid surface. The coefficient of friction between the contacting surfaces is 0.1.

The results from the end of the first step of this analysis are then imported into another Abaqus/Standard static analysis with the material state imported and the reference configuration not updated. During this import analysis the block is moved parallel to the rigid surface in exactly the same manner as in the second step of the first analysis. The results at the end of this import analysis should be identical to the results at the end of the first analysis.

The results for only the deformable block are then transferred from the end of the second analysis into a third Abaqus/Standard analysis by specifying only the element set that contains the deformable block on the data line of the import option. The material state is imported, and the reference configuration is updated. A new rigid surface is then defined along the top surface of the deformable block; new contact definitions for the interaction between the rigid surface and the top surface of the block are also specified. The bottom of the block is held fixed and the block is compressed by moving the newly defined rigid surface downward.

Results and discussion

From these tests it can be seen that rigid elements along with their rigid body reference node definitions and any contact conditions can be transferred from one Abaqus/Standard analysis to another. In addition, new rigid elements and contact conditions can be defined in an import analysis.

Input files

R2D2 element tests:

ss1_r2d2.inp	First Abaqus/Standard analysis.
ss2_r2d2.inp	Second Abaqus/Standard analysis.
ss3_r2d2.inp	Third Abaqus/Standard analysis.

R3D3 element tests:

ss1_r3d3.inp	First Abaqus/Standard analysis.
ss2_r3d3.inp	Second Abaqus/Standard analysis.
ss3_r3d3.inp	Third Abaqus/Standard analysis.

R3D4 element tests:

ss1_r3d4.inp	First Abaqus/Standard analysis.
ss1_r3d4_surf.inp	First Abaqus/Standard analysis (surface-to-surface contact).
ss2_r3d4.inp	Second Abaqus/Standard analysis.
ss2_r3d4_surf.inp	Second Abaqus/Standard analysis (surface-to-surface contact).
ss3_r3d4.inp	Third Abaqus/Standard analysis.

Transferring mass and rotary inertia elements

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Transferring results between Abaqus/Standard and Abaqus/Explicit

Elements tested

MASS ROTARYI

Problem description

The verification problems outlined in this section test the transfer of mass and rotary inertia elements between Abaqus/Standard and Abaqus/Explicit.

The models consist of two elastic trusses or beams that have identical properties except for material density. In the truss model mass elements are applied to the nodes of the truss with smaller density to make the total nodal mass the same as that of the truss with larger density. Similarly, in the beam model both mass and rotary inertia elements are applied to the nodes of the beam with smaller density to make the total nodal mass and rotary inertia about the beam axis the same for both beams.

The analysis consists of two steps. In the first step gravity loading is applied in a static Abaqus/Standard analysis or in an Abaqus/Explicit analysis. The results at the end of the first step are imported to either a dynamic Abaqus/Standard analysis or an Abaqus/Explicit analysis in which a pull or twist is applied at one end of the trusses or beams. The import analyses are performed with the states reinstated at the original configuration and with no states reinstated. Mass and rotary inertia elements do not have an associated elemental state, so only the states of the trusses or beams are imported in the original configuration.

Results and discussion

In the import analyses the nodal dynamic response of the elements in each model is the same. From these tests it is verified that mass and rotary inertia elements can be transferred between Abaqus/Standard and Abaqus/Explicit.

Input files

MASS element tests:

s_t2d2_mass.inp	First Abaqus/Standard analysis.
sx_x_t2d2_mass_n_y.inp	Second Abaqus/Explicit analysis, UPDATE=NO and STATE=YES.
sx_x_t2d2_mass_y_n.inp	Second Abaqus/Explicit analysis, UPDATE=YES and STATE=NO.
s_b31_mass_rotaryi.inp	First Abaqus/Standard analysis.
sx_x_b31_mass_n_y.inp	Second Abaqus/Explicit analysis, UPDATE=NO and STATE=YES.
sx_x_b31_mass_y_n.inp	Second Abaqus/Explicit analysis, UPDATE=YES and STATE=NO.
x_t2d2_mass.inp	First Abaqus/Explicit analysis.
xs_s_t2d2_mass_n_y.inp	Second Abaqus/Standard analysis, UPDATE=NO and STATE=YES.
xs_s_t2d2_mass_y_n.inp	Second Abaqus/Standard analysis, UPDATE=YES and STATE=NO.

ROTARYI element tests:

s_b31_mass_rotaryi.inp	First Abaqus/Standard analysis.
sx_x_b31_rotaryi_n_y.inp	Second Abaqus/Explicit analysis, UPDATE=NO and STATE=YES.
sx_x_b31_rotaryi_y_n.inp	Second Abaqus/Explicit analysis, UPDATE=YES and STATE=NO.

Transferring results from one Abaqus/Standard analysis to another Abaqus/Standard analysis

Elements tested

MASS

Problem description

The models consist of two elastic trusses that have identical properties except for material density. Mass elements are applied to the nodes of the truss with smaller density to make the total nodal mass the same as that of the truss with larger density.

The analysis consists of two steps. In the first step gravity loading is applied in a static analysis. The results at the end of the first static analysis are imported to a second dynamic analysis in which a pull is applied at one end of the trusses. The import analyses are performed with the states reinstated at the original configuration and with no states reinstated. Mass elements do not have an associated element state, so only the states of the trusses are imported in the original configuration.

Results and discussion

In the import analyses the nodal dynamic response of the elements in each model is the same. From these tests it is verified that mass elements can be transferred between Abaqus/Standard analyses.

Input files

MASS element tests:

s_t2d2_mass.inp	First Abaqus/Standard analysis.
ss_2_t2d2_mass_n_y.inp	Second Abaqus/Standard analysis, UPDATE=NO and STATE=YES.
ss_2_t2d2_mass_y_n.inp	Second Abaqus/Standard analysis, UPDATE=YES and STATE=NO.

Transferring results from one Abaqus/Explicit analysis to another Abaqus/Explicit analysis

Elements tested

MASS ROTARYI

Problem description

The models consist of two elastic trusses or beams that have identical properties except for material density. In the truss model mass elements are applied to the nodes of the truss with smaller density to make the total nodal mass the same as that of the truss with larger density. Similarly, in the beam model both mass and rotary inertia elements are applied to the nodes of the beam with smaller density to make the total nodal mass and rotary inertia about the beam axis the same for both beams.

The analysis consists of two steps. In the first step gravity loading is applied. The results at the end of the first analysis are imported to a second analysis in which a pull or twist is applied at one end of the trusses or beams. The import analyses are performed with the states reinstated at the original configuration and with no states reinstated. Mass and rotary inertia elements do not have an associated elemental state, so only the states of the trusses or beams are imported in the original configuration.

Results and discussion

In the import analyses the nodal dynamic response of the elements in each model is the same. From these tests it is verified that mass and rotary inertia elements can be transferred between Abaqus/Explicit analyses.

Input files

MASS element tests:

x_t2d2_mass.inp	First Abaqus/Explicit analysis.
xx_2_t2d2_mass_n_y.inp	Second Abaqus/Explicit analysis, UPDATE=NO and STATE=YES.
xx_2_t2d2_mass_y_n.inp	Second Abaqus/Explicit analysis, UPDATE=YES and STATE=NO.
x_b31_mass_rotaryi.inp	First Abaqus/Explicit analysis.
xx_2_b31_mass_n_y.inp	Second Abaqus/Explicit analysis, UPDATE=NO and STATE=YES.
xx_2_b31_mass_y_n.inp	Second Abaqus/Explicit analysis, UPDATE=YES and STATE=NO.

ROTARYI element tests:

x_b31_mass_rotaryi.inp	First Abaqus/Explicit analysis.
xx_2_b31_rotaryi_n_y.inp	Second Abaqus/Explicit analysis, UPDATE=NO and STATE=YES.
xx_2_b31_rotaryi_y_n.inp	Second Abaqus/Explicit analysis, UPDATE=YES and STATE=NO.

Transferring connector elements into Abaqus/Explicit

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

The verification of the import functionality of the connector elements is based on the analyses covered in *Connector element verification*. Typically, the Abaqus/Standard analyses covered in that section are imported and continued using Abaqus/Explicit. The results from Abaqus/Explicit at the point of import are compared with those from Abaqus/Standard. The analyses are continued by importing from the Abaqus/Explicit analyses to new Abaqus/Explicit analyses, and the results at the point of import are compared.

The connector elements can be imported from Abaqus/Standard or Abaqus/Explicit to Abaqus/Explicit. The import of connector elements to Abaqus/Standard is not available.

Damped free vibration with initial conditions

Elements tested

CONN2D2 CONN3D2

Problem description

See *Damped free vibration with initial conditions* for the problem description. The connector elements are imported, and any mass and rotary inertia elements in the model are redefined. Various types of connector sections are tested. The connector behavior includes elasticity and damping.

Results and discussion

The results demonstrate that the connector elements are transferred successfully from Abaqus/Standard to Abaqus/Explicit.

Input files

Abaqus/Standard input files

sx_s_conn_free_2d.inp	Connector elements in two dimensions.
sx_s_conn_free_3d.inp	Connector elements in three dimensions.
sx_s_conn_free_bushing.inp	Bushing element.

Abaqus/Explicit input files importing with UPDATE=NO and STATE=YES settings

sx_x_conn_free_2d_n_y.inp	Connector elements in two dimensions.
sx_x_conn_free_3d_n_y.inp	Connector elements in three dimensions.
sx_x_conn_free_bushing_n_y.inp	Bushing element.

Sinusoidal excitation of a damped spring-mass system

Elements tested

CONN2D2 CONN3D2

Problem description

See *Sinusoidal excitation of a damped spring-mass system* for the problem description. The connector elements are imported, and any mass and rotary inertia elements in the model are redefined. Various types of connector sections are tested. The connector behavior includes elasticity, damping, and friction.

Results and discussion

The results demonstrate that the connector elements are transferred successfully from Abaqus/Standard to Abaqus/Explicit.

Input files

Abaqus/Standard input files

sx_s_conn_force_2d.inp	Connection in two dimensions.
sxp_s_conn_force_2d.inp	Connection in two dimensions defined as assembly of part instances.
sx_s_conn_force_2d_fric.inp	Connection in two dimensions with friction.
sx_s_conn_force_3d.inp	Connection in three dimensions.
sxp_s_conn_force_3d.inp	Connection in three dimensions defined as assembly of part instances.
sx_s_conn_force_3d_fric.inp	Connection in three dimensions with friction.
sx_s_conn_force_bushing.inp	Bushing element.

Abaqus/Explicit input files importing with UPDATE=NO and STATE=YES settings

sx_x_conn_force_2d_n_y.inp	Connection in two dimensions.
sxp_x_conn_force_2d_n_y.inp	Connection in two dimensions defined as assembly of part instances.
sx_x_conn_force_2d_fric_n_y.inp	Connection in two dimensions with friction.
sx_x_conn_force_3d_n_y.inp	Connection in three dimensions.
sxp_x_conn_force_3d_n_y.inp	Connection in three dimensions defined as assembly of part instances.
sx_x_conn_force_3d_fric_n_y.inp	Connection in three dimensions with friction.

Abaqus/Explicit input files importing from the above Abaqus/Explicit analyses with UPDATE=NO and STATE=YES settings

xx2_conn_force_2d_n_y.inp	Connection in two dimensions.
xx2p_conn_force_2d_n_y.inp	Connection in two dimensions defined as assembly of part instances.
xx2_conn_force_2d_fric_n_y.inp	Connection in two dimensions with friction.
xx2_conn_force_3d_n_y.inp	Connection in three dimensions.

xx2p_conn_force_3d_n_y.inp	Connection in three dimensions defined as assembly of part instances.
xx2_conn_force_3d_fric_n_y.inp	Connection in three dimensions with friction.
xx2_conn_force_bushing_n_y.inp	Bushing element without friction.

Tests for special-purpose connectors

Elements tested

CONN3D2

Problem description

See *Tests for special-purpose connectors* for the problem descriptions covering the SLIPRING-type connectors with and without friction and the RETRACTOR-type connectors. The connector elements are imported, and any mass and rotary inertia elements in the model are redefined. The connector behavior includes elasticity, plasticity, and friction.

Results and discussion

The results demonstrate that the connector elements are transferred successfully from Abaqus/Standard to Abaqus/Explicit.

Input files

Abaqus/Standard input files

sx_s_misc_elasslipring_conn3d.inp	SLIPRING with linear elastic connector behavior.
sx_s_misc_slipring_conn3d.inp	SLIPRING connector with nonlinear elastic-plastic behavior with connector friction.
sx_s_misc_flowconverter_conn3d.inp	Retractor-type connection.

Abaqus/Explicit input files importing with UPDATE=NO and STATE=YES settings

sx_x_misc_elasslipring_conn3d_n_y.inp	SLIPRING with linear elastic connector behavior.
sx_x_misc_slipring_conn3d_n_y.inp	SLIPRING connector with nonlinear elastic-plastic behavior with connector friction.
sx_x_misc_flowconverter_conn3d_n_y.inp	Retractor-type connection.

Transferring hourglass forces

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D8R CPE4R CPS4R C3D10M CPE6M CPS6M

Problem description

The problem outlined in this section consists of a cantilever beam, as shown in *Figure 1*. This problem performs two tests. It tests the use of the direct-integration implicit dynamic procedure in the first step of Abaqus/Standard and the transfer of hourglass forces between Abaqus/Standard and Abaqus/Explicit. The following material definition is used for this model:

Young's modulus = 200×10^9

Poisson's ratio = 0.3

Density = 1000.

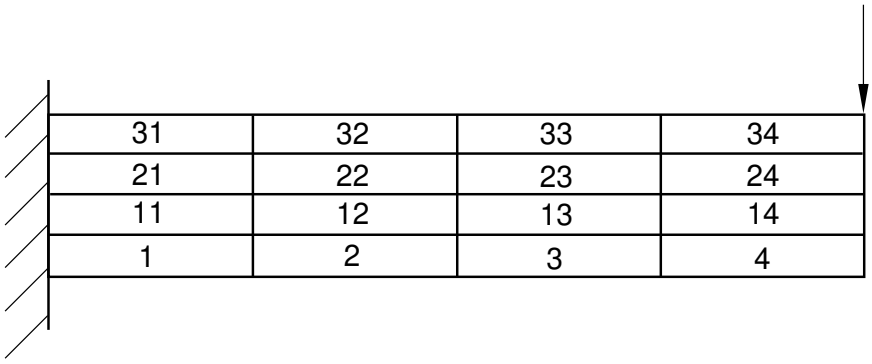


Figure 1: Model used for the bending test.

The analysis consists of four steps:

- The first step is performed in Abaqus/Standard using the direct-integration implicit dynamic procedure. In this step the beam is loaded by applying a displacement boundary condition at the tip of the beam, as shown in *Figure 1*.
- The results at the end of the first step are then imported into Abaqus/Explicit with the material state imported and the reference configuration not updated. In this step the displacements applied to the tip of the beam at the end of the first step are held fixed.
- The third step is also performed in Abaqus/Explicit. In this step the beam is displaced in the same direction as before by imposing an additional displacement boundary condition at its tip.
- The results at the end of the third step of the analysis are then imported into Abaqus/Standard with the material state imported and the reference configuration not updated. In this step displacement boundary conditions identical to those specified at the end of Step 1 are imposed at the tip of the cantilever beam. This step is performed using the direct-integration implicit dynamic procedure.

Verification tests of the enhanced hourglass control method are also included.

Step 2 of the analysis is also performed in Abaqus/Standard by continuing the analysis from the end of the first step. This allows for the comparison of the results between Abaqus/Explicit and Abaqus/Standard.

Results and discussion

Figure 2 shows the time history of the stress component, σ_{11}, at the integration point of element 1 (CPE4R elements are used in this test). It can be seen that the stress state in Step 2 obtained from the Abaqus/Explicit analysis is nearly the same as that obtained from the Abaqus/Standard analysis. In all cases results similar to the ones shown in *Figure 2* are obtained.

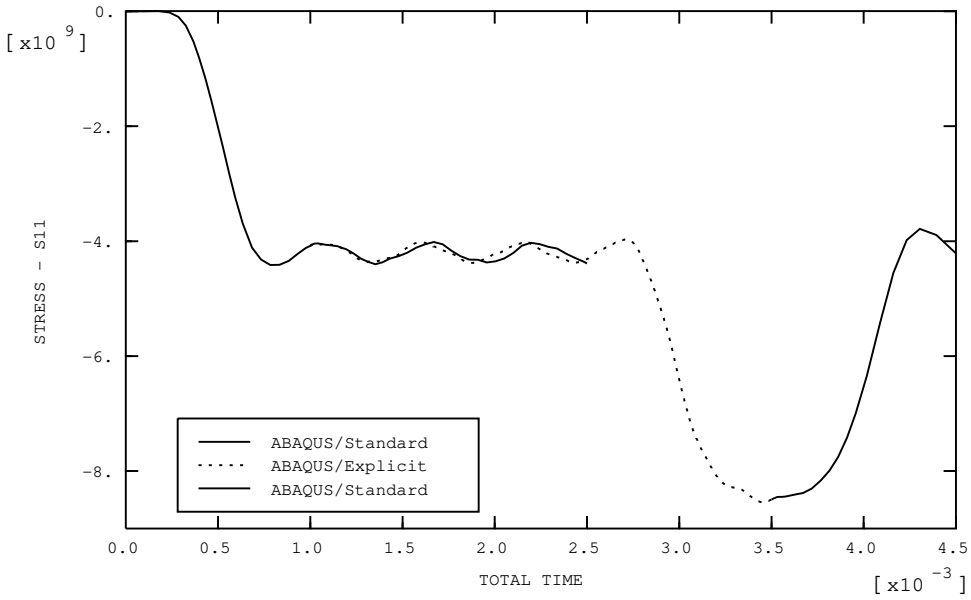


Figure 2: Time history of the stress component, σ_{11}.

Input files

C3D8R element tests:

sx_s_c3d8r_hg.inp	First Abaqus/Standard analysis.
sx_x_c3d8r_hg.inp	Abaqus/Explicit analysis.
xs_s_c3d8r_hg.inp	Second Abaqus/Standard analysis.
sx_s_c3d8r_hg_enhg.inp	First Abaqus/Standard analysis with enhanced hourglass control.
sx_x_c3d8r_hg_enhg.inp	Abaqus/Explicit analysis with enhanced hourglass control.
xs_s_c3d8r_hg_enhg.inp	Second Abaqus/Standard analysis with enhanced hourglass control.

CPE4R element tests:

sx_s_cpe4r_hg.inp	First Abaqus/Standard analysis.
sx_x_cpe4r_hg.inp	Abaqus/Explicit analysis.
xs_s_cpe4r_hg.inp	Second Abaqus/Standard analysis.
sx_s_cpe4r_hg_enhg.inp	First Abaqus/Standard analysis with enhanced hourglass control.
sx_x_cpe4r_hg_enhg.inp	Abaqus/Explicit analysis with enhanced hourglass control.
xs_s_cpe4r_hg_enhg.inp	Second Abaqus/Standard analysis with enhanced hourglass control.

CPS4R element tests:

sx_s_cps4r_hg.inp	First Abaqus/Standard analysis.
sx_x_cps4r_hg.inp	Abaqus/Explicit analysis.
xs_s_cps4r_hg.inp	Second Abaqus/Standard analysis.
sx_s_cps4r_hg_enhg.inp	First Abaqus/Standard analysis with enhanced hourglass control.
sx_x_cps4r_hg_enhg.inp	Abaqus/Explicit analysis with enhanced hourglass control.
xs_s_cps4r_hg_enhg.inp	Second Abaqus/Standard analysis with enhanced hourglass control.

C3D10M element tests:

sx_s_c3d10m_hg_enhg.inp	First Abaqus/Standard analysis with enhanced hourglass control.
sx_x_c3d10m_hg_enhg.inp	Abaqus/Explicit analysis with enhanced hourglass control.
xs_s_c3d10m_hg_enhg.inp	Second Abaqus/Standard analysis with enhanced hourglass control.

CPE6M element tests:

sx_s_cpe6m_hg_enhg.inp	First Abaqus/Standard analysis with enhanced hourglass control.
sx_x_cpe6m_hg_enhg.inp	Abaqus/Explicit analysis with enhanced hourglass control.
xs_s_cpe6m_hg_enhg.inp	Second Abaqus/Standard analysis with enhanced hourglass control.

CPS6M element tests:

sx_s_cps6m_hg_enhg.inp	First Abaqus/Standard analysis with enhanced hourglass control.
sx_x_cps6m_hg_enhg.inp	Abaqus/Explicit analysis with enhanced hourglass control.
xs_s_cps6m_hg_enhg.inp	Second Abaqus/Standard analysis with enhanced hourglass control.

CPE4R element tests with the reference configuration updated:

sx_s_cpe4r_hg.inp	First Abaqus/Standard analysis.
sx_x_cpe4r_hg_y.inp	Abaqus/Explicit analysis.
xs_s_cpe4r_hg_y.inp	Second Abaqus/Standard analysis.
sx_s_cpe4r_hg_enhg.inp	First Abaqus/Standard analysis with enhanced hourglass control.
sx_x_cpe4r_hg_y_enhg.inp	Abaqus/Explicit analysis with enhanced hourglass control.
xs_s_cpe4r_hg_y_enhg.inp	Second Abaqus/Standard analysis with enhanced hourglass control.

Changing the material definition during import

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

CPE4R

Problem description

The problem considered here demonstrates the ability to change the material definition and continue the analysis after import. An elastic-plastic material with Mises yield criterion is used in the Abaqus/Standard analysis. The analysis is continued in Abaqus/Explicit by introducing a ductile failure model using the shear failure model. The square cross-section of a prismatic bar under transverse biaxial tensile loading is modeled using CPE4R elements. Due to symmetry of the geometry and the loading, only one-quarter of the domain is modeled, as shown in *Figure 1*.

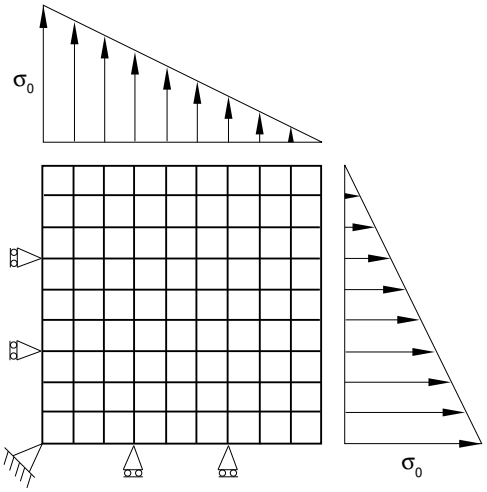


Figure 1: Model for verification of change of material.

In the Abaqus/Standard analysis the object is loaded so that part of the domain begins to yield. The loading is continued in the Abaqus/Explicit analysis so that the plastic strains reach into the failure regime. The results of the Abaqus/Explicit analysis are imported back into Abaqus/Standard to verify that the failed elements are not imported. The material properties used in Abaqus/Standard are as follows:

Young's modulus = 207.8×10^9

Poisson's ratio = 0.3

Density = 7800.

Yield stress = $1220. \times 10^6$

Flow stress = $1440. \times 10^6$ when $\bar{\varepsilon}^{pl} = 1.0$

In Abaqus/Explicit ductile failure is specified so that the failure starts when the equivalent plastic strain reaches 0.8 and the complete failure is reached when the equivalent plastic strain reaches a value of unity. The load is specified in Abaqus/Standard so that the maximum traction, σ_0, is 2.5 times the initial yield stress; and in Abaqus/Explicit it is increased to a value of 4 times the initial yield stress. The material state is imported, and the reference configuration is not updated.

Results and discussion

This problem demonstrates the flexibility in changing the material definition judiciously and continuing the analysis after import. The stresses, strains, and energy quantities such as recoverable elastic strain energy are found to be continuous across the Abaqus/Standard and Abaqus/Explicit analyses. Failed elements are not imported from Abaqus/Explicit to Abaqus/Standard.

Input files

sx_s_cpe4r_f.inp	First Abaqus/Standard analysis.
sx_x_cpe4r_f_n_y.inp	Abaqus/Explicit analysis.
xs_s_cpe4r_f_n_y.inp	Second Abaqus/Standard analysis.

Transferring results with plasticity

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

B21 B22 B31 B32 C3D4 C3D6 C3D8R CAX3 CAX4R
CPE3 CPE4R CPS3 CPS4R M3D3 M3D4R SAX1 S3R S4R S4 T2D2 T3D2
C3D10M CAX6M CPE6M CPS6M

Problem description

The verification tests in this section consist of one-element models that are subjected to monotonically increasing tensile loads in a sequential import analysis. The sequence of tests involves transferring results from Abaqus/Explicit to Abaqus/Standard and then back to Abaqus/Explicit. An elastic-plastic material with Mises yield criterion is used in the analyses. Two sets of problems are tested: one with isotropic hardening and the other with combined isotropic/kinematic hardening.

The following material properties are used (the units are not important):

Elasticity

Young's modulus, $E = 200.0 \times 10^3$

Poisson's ratio, $\nu = 0.3$

Plasticity for isotropic hardening case

Yield stress	Plastic strain
200.	0.0000
220.	0.001
220.	0.003

Plasticity for combined isotropic/kinematic hardening case

Initial yield stress: $\sigma_0 = 200.0$

Isotropic hardening parameter, $Q_\infty = 2000.0$

Isotropic hardening parameter, $b = 0.25$

Kinematic hardening parameter, $C = 2.222 \times 10^4$

Kinematic hardening parameter, $\gamma = 34.65$

Plasticity for combined isotropic/kinematic hardening with multiple backstresses case

Initial yield stress: σ_0 = 200.0

Isotropic hardening parameter, Q_∞ = 2000.0

Isotropic hardening parameter, b = 0.25

Kinematic hardening parameter, C_1 = 1.111×10^4

Kinematic hardening parameter, γ_1 = 34.65

Kinematic hardening parameter, C_2 = 5.555×10^3

Kinematic hardening parameter, γ_2 = 34.65

Kinematic hardening parameter, C_3 = 5.555×10^3

Kinematic hardening parameter, γ_3 = 34.65

A verification test is also conducted for adiabatic Mises plasticity with isotropic hardening. C3D8R elements are used in the analysis. The following material properties are used (the units are not important):

Elasticity

Young's modulus, E = 30.0×10^6

Poisson's ratio, ν = 0.3

Plasticity (isotropic hardening)

Yield stress	Plastic strain	Temperature
30.0E3	0.000	0.0
50.0E3	0.200	0.0
50.0E3	2.000	0.0
3.0E3	0.000	100.0
5.0E3	0.200	100.0
5.0E3	2.000	100.0

Other properties

Density, μ = 1000.0

Specific heat, c = 0.4

Inelastic heat fraction, 0.5

Verification tests are also included for some first-order reduced-integration elements with enhanced hourglass control.

Results and discussion

The results demonstrate that the plasticity material model is transferred successfully between Abaqus/Explicit and Abaqus/Standard.

Input files

B21 element tests:Isotropic hardening:Combined hardening:Combined hardening (multiple backstresses):

xs_x_b21_t_pl.inp	First Abaqus/Explicit analysis.
xs_s_b21_t_pl.inp	Abaqus/Standard analysis.
sx_x_b21_t_pl.inp	Second Abaqus/Explicit analysis.
xs_x_b21_t_plch.inp	First Abaqus/Explicit analysis.
xs_s_b21_t_plch.inp	Abaqus/Standard analysis.
sx_x_b21_t_plch.inp	Second Abaqus/Explicit analysis.
xs_x_b21_t_plchmb.inp	First Abaqus/Explicit analysis.
xs_s_b21_t_plchmb.inp	Abaqus/Standard analysis.
sx_x_b21_t_plchmb.inp	Second Abaqus/Explicit analysis.

B22 element tests:Isotropic hardening:Combined hardening:Combined hardening (multiple backstresses):

xs_x_b22_t_pl.inp	First Abaqus/Explicit analysis.
xs_s_b22_t_pl.inp	Abaqus/Standard analysis.
sx_x_b22_t_pl.inp	Second Abaqus/Explicit analysis.
xs_x_b22_t_plch.inp	First Abaqus/Explicit analysis.
xs_s_b22_t_plch.inp	Abaqus/Standard analysis.
sx_x_b22_t_plch.inp	Second Abaqus/Explicit analysis.
xs_x_b22_t_plchmb.inp	First Abaqus/Explicit analysis.
xs_s_b22_t_plchmb.inp	Abaqus/Standard analysis.
sx_x_b22_t_plchmb.inp	Second Abaqus/Explicit analysis.

B31 element tests:Isotropic hardening:

xs_x_b31_t_pl.inp	First Abaqus/Explicit analysis.
xs_s_b31_t_pl.inp	Abaqus/Standard analysis.
sx_x_b31_t_pl.inp	Second Abaqus/Explicit analysis.

Combined hardening not available for B31 elements in Abaqus/Standard.B32 element tests:Isotropic hardening:

xs_x_b32_t_pl.inp	First Abaqus/Explicit analysis.
xs_s_b32_t_pl.inp	Abaqus/Standard analysis.
sx_x_b32_t_pl.inp	Second Abaqus/Explicit analysis.

Combined hardening not available for B32 elements in Abaqus/Standard.C3D4 element tests:Isotropic hardening:Combined hardening:Combined hardening (multiple backstresses):

xs_x_c3d4_pl.inp	First Abaqus/Explicit analysis.
xs_s_c3d4_pl.inp	Abaqus/Standard analysis.
sx_x_c3d4_pl.inp	Second Abaqus/Explicit analysis.
xs_x_c3d4_plch.inp	First Abaqus/Explicit analysis.
xs_s_c3d4_plch.inp	Abaqus/Standard analysis.

sx_x_c3d4_plch.inp	Second Abaqus/Explicit analysis.
xs_x_c3d4_plchmb.inp	First Abaqus/Explicit analysis.
xs_s_c3d4_plchmb.inp	Abaqus/Standard analysis.
sx_x_c3d4_plchmb.inp	Second Abaqus/Explicit analysis.

C3D6 element tests:Isotropic hardening:Combined hardening:Combined hardening (multiple backstresses):

xs_x_c3d6_pl.inp	First Abaqus/Explicit analysis.
xs_s_c3d6_pl.inp	Abaqus/Standard analysis.
sx_x_c3d6_pl.inp	Second Abaqus/Explicit analysis.
xs_x_c3d6_plch.inp	First Abaqus/Explicit analysis.
xs_s_c3d6_plch.inp	Abaqus/Standard analysis.
sx_x_c3d6_plch.inp	Second Abaqus/Explicit analysis.
xs_x_c3d6_plchmb.inp	First Abaqus/Explicit analysis.
xs_s_c3d6_plchmb.inp	Abaqus/Standard analysis.
sx_x_c3d6_plchmb.inp	Second Abaqus/Explicit analysis.

C3D8R element tests:Isotropic hardening:Combined hardening:Combined hardening (multiple backstresses):

xs_x_c3d8r_pl.inp	First Abaqus/Explicit analysis.
xs_s_c3d8r_pl.inp	Abaqus/Standard analysis.
sx_x_c3d8r_pl.inp	Second Abaqus/Explicit analysis.
xs_x_c3d8r_plch.inp	First Abaqus/Explicit analysis.
xs_s_c3d8r_plch.inp	Abaqus/Standard analysis.
sx_x_c3d8r_plch.inp	Second Abaqus/Explicit analysis.
xs_x_c3d8r_plchmb.inp	First Abaqus/Explicit analysis.
xs_s_c3d8r_plchmb.inp	Abaqus/Standard analysis.
sx_x_c3d8r_plchmb.inp	Second Abaqus/Explicit analysis.

CAX3 element tests:Isotropic hardening:Combined hardening:Combined hardening (multiple backstresses):

xs_x_cax3_pl.inp	First Abaqus/Explicit analysis.
xs_s_cax3_pl.inp	Abaqus/Standard analysis.
sx_x_cax3_pl.inp	Second Abaqus/Explicit analysis.
xs_x_cax3_plch.inp	First Abaqus/Explicit analysis.
xs_s_cax3_plch.inp	Abaqus/Standard analysis.
sx_x_cax3_plch.inp	Second Abaqus/Explicit analysis.
xs_x_cax3_plchmb.inp	First Abaqus/Explicit analysis.
xs_s_cax3_plchmb.inp	Abaqus/Standard analysis.
sx_x_cax3_plchmb.inp	Second Abaqus/Explicit analysis.

CAX4R element tests:Isotropic hardening:Combined hardening:Combined hardening (multiple backstresses):

xs_x_cax4r_pl.inp	First Abaqus/Explicit analysis.
xs_s_cax4r_pl.inp	Abaqus/Standard analysis.
sx_x_cax4r_pl.inp	Second Abaqus/Explicit analysis.
xs_x_cax4r_plch.inp	First Abaqus/Explicit analysis.
xs_s_cax4r_plch.inp	Abaqus/Standard analysis.
sx_x_cax4r_plch.inp	Second Abaqus/Explicit analysis.

xs_x_cax4r_plchmb.inp	First Abaqus/Explicit analysis.
xs_s_cax4r_plchmb.inp	Abaqus/Standard analysis.
sx_x_cax4r_plchmb.inp	Second Abaqus/Explicit analysis.

CPE3 element tests:Isotropic hardening:Combined hardening:Combined hardening (multiple backstresses):

xs_x_cpe3_pl.inp	First Abaqus/Explicit analysis.
xs_s_cpe3_pl.inp	Abaqus/Standard analysis.
sx_x_cpe3_pl.inp	Second Abaqus/Explicit analysis.
xs_x_cpe3_plch.inp	First Abaqus/Explicit analysis.
xs_s_cpe3_plch.inp	Abaqus/Standard analysis.
sx_x_cpe3_plch.inp	Second Abaqus/Explicit analysis.
xs_x_cpe3_plchmb.inp	First Abaqus/Explicit analysis.
xs_s_cpe3_plchmb.inp	Abaqus/Standard analysis.
sx_x_cpe3_plchmb.inp	Second Abaqus/Explicit analysis.

CPE4R element tests:Isotropic hardening:Combined hardening:Combined hardening (multiple backstresses):

xs_x_cpe4r_pl.inp	First Abaqus/Explicit analysis.
xs_x_cpe4r_pl_enhg.inp	First Abaqus/Explicit analysis with enhanced hourglass control.
xs_s_cpe4r_pl.inp	Abaqus/Standard analysis.
xs_s_cpe4r_pl_enhg.inp	Abaqus/Standard analysis with enhanced hourglass control.
sx_x_cpe4r_pl.inp	Second Abaqus/Explicit analysis.
sx_x_cpe4r_pl_enhg.inp	Second Abaqus/Explicit analysis with enhanced hourglass control.
xs_x_cpe4r_plch.inp	First Abaqus/Explicit analysis.
xs_s_cpe4r_plch.inp	Abaqus/Standard analysis.
sx_x_cpe4r_plch.inp	Second Abaqus/Explicit analysis.
xs_x_cpe4r_plchmb.inp	First Abaqus/Explicit analysis.
xs_s_cpe4r_plchmb.inp	Abaqus/Standard analysis.
sx_x_cpe4r_plchmb.inp	Second Abaqus/Explicit analysis.

CPS3 element tests:Isotropic hardening:Combined hardening:Combined hardening (multiple backstresses):

xs_x_cps3_pl.inp	First Abaqus/Explicit analysis.
xs_s_cps3_pl.inp	Abaqus/Standard analysis.
sx_x_cps3_pl.inp	Second Abaqus/Explicit analysis.
xs_x_cps3_plch.inp	First Abaqus/Explicit analysis.
xs_s_cps3_plch.inp	Abaqus/Standard analysis.
sx_x_cps3_plch.inp	Second Abaqus/Explicit analysis.
xs_x_cps3_plchmb.inp	First Abaqus/Explicit analysis.
xs_s_cps3_plchmb.inp	Abaqus/Standard analysis.
sx_x_cps3_plchmb.inp	Second Abaqus/Explicit analysis.

CPS4R element tests:Isotropic hardening:Combined hardening:Combined hardening (multiple backstresses):

xs_x_cps4r_pl.inp	First Abaqus/Explicit analysis.
xs_s_cps4r_pl.inp	Abaqus/Standard analysis.
sx_x_cps4r_pl.inp	Second Abaqus/Explicit analysis.

xs_x_cps4r_plch.inp	First Abaqus/Explicit analysis.
xs_s_cps4r_plch.inp	Abaqus/Standard analysis.
sx_x_cps4r_plch.inp	Second Abaqus/Explicit analysis.
xs_x_cps4r_plchmb.inp	First Abaqus/Explicit analysis.
xs_s_cps4r_plchmb.inp	Abaqus/Standard analysis.
sx_x_cps4r_plchmb.inp	Second Abaqus/Explicit analysis.

C3D10M element tests:Isotropic hardening:Combined hardening:Combined hardening (multiple backstresses):

xs_x_c3d10m_pl.inp	First Abaqus/Explicit analysis.
xs_s_c3d10m_pl.inp	Abaqus/Standard analysis.
sx_x_c3d10m_pl.inp	Second Abaqus/Explicit analysis.
xs_x_c3d10m_plch.inp	First Abaqus/Explicit analysis.
xs_s_c3d10m_plch.inp	Abaqus/Standard analysis.
sx_x_c3d10m_plch.inp	Second Abaqus/Explicit analysis.
xs_x_c3d10m_plchmb.inp	First Abaqus/Explicit analysis.
xs_s_c3d10m_plchmb.inp	Abaqus/Standard analysis.
sx_x_c3d10m_plchmb.inp	Second Abaqus/Explicit analysis.

CAX6M element tests:Isotropic hardening:Combined hardening:Combined hardening (multiple backstresses):

xs_x_cax6m_pl.inp	First Abaqus/Explicit analysis.
xs_s_cax6m_pl.inp	Abaqus/Standard analysis.
sx_x_cax6m_pl.inp	Second Abaqus/Explicit analysis.
xs_x_cax6m_plch.inp	First Abaqus/Explicit analysis.
xs_s_cax6m_plch.inp	Abaqus/Standard analysis.
sx_x_cax6m_plch.inp	Second Abaqus/Explicit analysis.
xs_x_cax6m_plchmb.inp	First Abaqus/Explicit analysis.
xs_s_cax6m_plchmb.inp	Abaqus/Standard analysis.
sx_x_cax6m_plchmb.inp	Second Abaqus/Explicit analysis.

CPE6M element tests:Isotropic hardening:Combined hardening:Combined hardening (multiple backstresses):

xs_x_cpe6m_pl.inp	First Abaqus/Explicit analysis.
xs_s_cpe6m_pl.inp	Abaqus/Standard analysis.
sx_x_cpe6m_pl.inp	Second Abaqus/Explicit analysis.
xs_x_cpe6m_plch.inp	First Abaqus/Explicit analysis.
xs_s_cpe6m_plch.inp	Abaqus/Standard analysis.
sx_x_cpe6m_plch.inp	Second Abaqus/Explicit analysis.
xs_x_cpe6m_plchmb.inp	First Abaqus/Explicit analysis.
xs_s_cpe6m_plchmb.inp	Abaqus/Standard analysis.
sx_x_cpe6m_plchmb.inp	Second Abaqus/Explicit analysis.

CPS6M element tests:Isotropic hardening:Combined hardening:Combined hardening (multiple backstresses):

xs_x_cps6m_pl.inp	First Abaqus/Explicit analysis.
xs_s_cps6m_pl.inp	Abaqus/Standard analysis.
sx_x_cps6m_pl.inp	Second Abaqus/Explicit analysis.
xs_x_cps6m_plch.inp	First Abaqus/Explicit analysis.

xs_s_cps6m_plch.inp	Abaqus/Standard analysis.
sx_x_cps6m_plch.inp	Second Abaqus/Explicit analysis.
xs_x_cps6m_plchmb.inp	First Abaqus/Explicit analysis.
xs_s_cps6m_plchmb.inp	Abaqus/Standard analysis.
sx_x_cps6m_plchmb.inp	Second Abaqus/Explicit analysis.

M3D3 element tests:Isotropic hardening:Combined hardening:Combined hardening (multiple backstresses):

xs_x_m3d3_pl.inp	First Abaqus/Explicit analysis.
xs_s_m3d3_pl.inp	Abaqus/Standard analysis.
sx_x_m3d3_pl.inp	Second Abaqus/Explicit analysis.
xs_x_m3d3_plch.inp	First Abaqus/Explicit analysis.
xs_s_m3d3_plch.inp	Abaqus/Standard analysis.
sx_x_m3d3_plch.inp	Second Abaqus/Explicit analysis.
xs_x_m3d3_plchmb.inp	First Abaqus/Explicit analysis.
xs_s_m3d3_plchmb.inp	Abaqus/Standard analysis.
sx_x_m3d3_plchmb.inp	Second Abaqus/Explicit analysis.

M3D4R element tests:Isotropic hardening:Combined hardening:Combined hardening (multiple backstresses):

xs_x_m3d4r_pl.inp	First Abaqus/Explicit analysis.
xs_x_m3d4r_pl_enhg.inp	First Abaqus/Explicit analysis with enhanced hourglass control.
xs_s_m3d4r_pl.inp	Abaqus/Standard analysis.
xs_s_m3d4r_pl_enhg.inp	Abaqus/Standard analysis with enhanced hourglass control.
sx_x_m3d4r_pl.inp	Second Abaqus/Explicit analysis.
sx_x_m3d4r_pl_enhg.inp	Second Abaqus/Explicit analysis with enhanced hourglass control.
xs_x_m3d4r_plch.inp	First Abaqus/Explicit analysis.
xs_s_m3d4r_plch.inp	Abaqus/Standard analysis.
sx_x_m3d4r_plch.inp	Second Abaqus/Explicit analysis.
xs_x_m3d4r_plchmb.inp	First Abaqus/Explicit analysis.
xs_s_m3d4r_plchmb.inp	Abaqus/Standard analysis.
sx_x_m3d4r_plchmb.inp	Second Abaqus/Explicit analysis.

SAX1 element tests:Isotropic hardening:Combinedi hardening:Combinedi hardening (multiple backstresses):

xs_x_sax_pl.inp	First Abaqus/Explicit analysis.
xs_s_sax_pl.inp	Abaqus/Standard analysis.
sx_x_sax_pl.inp	Second Abaqus/Explicit analysis.
xs_x_sax_plch.inp	First Abaqus/Explicit analysis.
xs_s_sax_plch.inp	Abaqus/Standard analysis.
sx_x_sax_plch.inp	Second Abaqus/Explicit analysis.
xs_x_sax_plchmb.inp	First Abaqus/Explicit analysis.
xs_s_sax_plchmb.inp	Abaqus/Standard analysis.
sx_x_sax_plchmb.inp	Second Abaqus/Explicit analysis.

S3R element tests:Isotropic hardening:Combined hardening:Combined hardening (multiple backstresses):

xs_x_s3r_pl.inp	First Abaqus/Explicit analysis.

xs_s_s3r_pl.inp	Abaqus/Standard analysis.
sx_x_s3r_pl.inp	Second Abaqus/Explicit analysis.
xs_x_s3r_plch.inp	First Abaqus/Explicit analysis.
xs_s_s3r_plch.inp	Abaqus/Standard analysis.
sx_x_s3r_plch.inp	Second Abaqus/Explicit analysis.
xs_x_s3r_plchmb.inp	First Abaqus/Explicit analysis.
xs_s_s3r_plchmb.inp	Abaqus/Standard analysis.
sx_x_s3r_plchmb.inp	Second Abaqus/Explicit analysis.

S4R element tests:Isotropic hardening:Combined hardening:Combined hardening (multiple backstresses):

xs_x_s4r_pl.inp	First Abaqus/Explicit analysis.
xs_s_s4r_pl.inp	Abaqus/Standard analysis.
sx_x_s4r_pl.inp	Second Abaqus/Explicit analysis.
xs_x_s4r_plch.inp	First Abaqus/Explicit analysis.
xs_s_s4r_plch.inp	Abaqus/Standard analysis.
sx_x_s4r_plch.inp	Second Abaqus/Explicit analysis.
xs_x_s4r_plchmb.inp	First Abaqus/Explicit analysis.
xs_s_s4r_plchmb.inp	Abaqus/Standard analysis.
sx_x_s4r_plchmb.inp	Second Abaqus/Explicit analysis.

S4 element tests:Combined hardening:Combined hardening (multiple backstresses):

xs_x_s4_plch.inp	First Abaqus/Explicit analysis.
xs_s_s4_plch.inp	Abaqus/Standard analysis.
sx_x_s4_plch.inp	Second Abaqus/Explicit analysis.
xs_x_s4_plchmb.inp	First Abaqus/Explicit analysis.
xs_s_s4_plchmb.inp	Abaqus/Standard analysis.
sx_x_s4_plchmb.inp	Second Abaqus/Explicit analysis.

T2D2 element tests:Isotropic hardening:Combined hardening:Combined hardening (multiple backstresses):

xs_x_t2d2_pl.inp	First Abaqus/Explicit analysis.
xs_s_t2d2_pl.inp	Abaqus/Standard analysis.
sx_x_t2d2_pl.inp	Second Abaqus/Explicit analysis.
xs_x_t2d2_plch.inp	First Abaqus/Explicit analysis.
xs_s_t2d2_plch.inp	Abaqus/Standard analysis.
sx_x_t2d2_plch.inp	Second Abaqus/Explicit analysis.
xs_x_t2d2_plchmb.inp	First Abaqus/Explicit analysis.
xs_s_t2d2_plchmb.inp	Abaqus/Standard analysis.
sx_x_t2d2_plchmb.inp	Second Abaqus/Explicit analysis.

T3D2 element tests:Isotropic hardening:Combined hardening:Combined hardening (multiple backstresses):

xs_x_t3d2_pl.inp	First Abaqus/Explicit analysis.
xs_s_t3d2_pl.inp	Abaqus/Standard analysis.
sx_x_t3d2_pl.inp	Second Abaqus/Explicit analysis.
xs_x_t3d2_plch.inp	First Abaqus/Explicit analysis.
xs_s_t3d2_plch.inp	Abaqus/Standard analysis.

sx_x_t3d2_plch.inp	Second Abaqus/Explicit analysis.
xs_x_t3d2_plchmb.inp	First Abaqus/Explicit analysis.
xs_s_t3d2_plchmb.inp	Abaqus/Standard analysis.
sx_x_t3d2_plchmb.inp	Second Abaqus/Explicit analysis.

Adiabatic Mises plasticity tests with the C3D8R element:

xs_x_c3d8r_ad.inp	First Abaqus/Explicit analysis.
xs_s_c3d8r_ad.inp	Abaqus/Standard analysis.
sx_x_c3d8r_ad.inp	Second Abaqus/Explicit analysis.

Transferring results with damage

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D8R CPS4R S4R

Problem description

The verification tests in this section consist of one-element models that are subjected to monotonically increasing tensile loads in sequential import analyses. Two sequences of tests are performed. The first sequence involves transferring results from Abaqus/Explicit to Abaqus/Standard and then again to Abaqus/Standard; the second involves transferring results from Abaqus/Standard to Abaqus/Explicit only. In the analyses an elastic-plastic material with Mises yield criterion is used in conjunction with ductile, shear, FLD, FLSD, and MSFLD damage initiation criteria and displacement or energy-based damage evolution laws.

The following material properties are used (the units are not important):

Material 1:

Elasticity

Young's modulus, $E = 2.0 \times 10^{11}$

Poisson's ratio, $\nu = 0.33$

Density

$\rho = 2000.0$

Plasticity (isotropic hardening)

Yield stress	Plastic strain
2.0×10^8	0.0
6.0×10^8	2.0

Ductile damage initiation properties

Equivalent plastic strain at damage initiation	Stress triaxiality
1.8	–0.5
1.5	–0.1
1.0	0.0
0.90	0.1
0.80	0.3

Equivalent plastic strain at damage initiation	Stress triaxiality
0.50	0.6
0.30	1.0

Material 2:

Elasticity

Young's modulus, $E = 2.0 \times 10^{11}$

Poisson's ratio, $\nu = 0.33$

Density

$\rho = 2000.0$

Plasticity (isotropic hardening)

Yield stress	Plastic strain
2.0×10^8	0.0
6.0×10^8	2.0

Ductile damage initiation properties

Equivalent plastic strain at damage initiation	Shear stress ratio	Strain rate
0.6	−10.	0.001
0.6	1.4	0.001
1.0	1.8	0.001
1.6	2.0	0.001
2.3	2.5	0.001
2.4	10.	0.001

Material 3:

Elasticity

Young's modulus, $E = 2.0 \times 10^{11}$

Poisson's ratio, $\nu = 0.33$

Density

$\rho = 2000.0$

Plasticity (isotropic hardening)

Yield stress	Plastic strain
2.0×10^8	0.0
4.0×10^8	2.0

FLD damage initiation properties

Major principal strain	Minor principal strain
0.30	–0.2
0.20	–0.1
0.15	0.0
0.25	0.1
0.40	0.2

Material 4:

Elasticity

Young's modulus, $E = 2.0 \times 10^{11}$

Poisson's ratio, $\nu = 0.33$

Density

$\rho = 2000.0$

Plasticity (isotropic hardening)

Yield stress	Plastic strain
2.0×10^{8}	0.0
4.0×10^{8}	2.0

FLSD damage initiation properties

Major principal strain	Minor principal strain
3.0×10^{8}	2.0×10^{8}
3.5×10^{8}	3.0×10^{8}
4.0×10^{8}	3.5×10^{8}
4.2×10^{8}	4.0×10^{8}

Material 5:

Elasticity

Young's modulus, $E = 2.0 \times 10^{11}$

Poisson's ratio, $\nu = 0.33$

Density

$\rho = 2000.0$

Plasticity (isotropic hardening)

Yield stress	Plastic strain
3.0×10^8	0.0
4.0×10^8	2.0

MSFLD damage initiation properties

Major principal strain	Minor principal strain
0.30	–0.2
0.15	0.0
0.20	0.1
0.25	0.2
0.30	0.4
0.40	0.6
0.60	0.7

Material 6:

Damage evolution properties for the evolution law based on equivalent plastic displacement with linear softening

Effective plasticdisplacement at failure
1.0

All other material parameters are identical to those specified for Material 1.

Material 7:

Damage evolution properties for the evolution law based on equivalent plastic displacement with tabular softening

Damage variable	Effective plasticdisplacement at failure
0.0	0.0
1.0	1.0

All other material parameters are identical to those specified for Material 1.

Material 8:

Damage evolution properties for the evolution law based on equivalent plastic displacement with exponential softening

Effective plasticdisplacement at failure	Exponential lawparameter
0.25	1.0

All other material parameters are identical to those specified for Material 1.

Material 9:

Damage evolution properties for the evolution law based on energy dissipation with linear softening

Fracture energy
4.0×10^8

All other material parameters are identical to those specified for Material 1.

Material 10:

Damage evolution properties for the evolution law based on energy dissipation with exponential softening

Fracture energy
1.0×10^8

All other material parameters are identical to those specified for Material 1.

Results and discussion

The results demonstrate that the plasticity material model with a damage initiation criterion and a damage evolution law is transferred successfully between Abaqus/Explicit and Abaqus/Standard.

Input files

Ductile damage initiation (Material 1):

xs_x_ductile.inp	First Abaqus/Explicit analysis.
xs_s_ductile.inp	Abaqus/Standard analysis.
ss_s_ductile.inp	Second Abaqus/Standard analysis.
sx_s_ductile.inp	Base problem for Abaqus/Standard to Abaqus/Explicit import.
sx_x_ductile.inp	Abaqus/Explicit analysis, imported from sx_s_ductile.inp.

Shear damage initiation (Material 2):

xs_x_shear.inp	First Abaqus/Explicit analysis.
xs_s_shear.inp	Abaqus/Standard analysis.
ss_s_shear.inp	Second Abaqus/Standard analysis.
sx_s_shear.inp	Base problem for Abaqus/Standard to Abaqus/Explicit import.
sx_x_shear.inp	Abaqus/Explicit analysis, imported from sx_s_shear.inp.

FLD damage initiation (Material 3):

xs_x_fld.inp	First Abaqus/Explicit analysis.
xs_x_fld.inp	Abaqus/Standard analysis.
ss_s_fld.inp	Second Abaqus/Standard analysis.
sx_s_fld.inp	Base problem for Abaqus/Standard to Abaqus/Explicit import.
sx_x_fld.inp	Abaqus/Explicit analysis, imported from sx_s_fld.inp.

FLSD damage initiation (Material 4):

xs_x_flsd.inp	First Abaqus/Explicit analysis.

xs_s_flsd.inp	Abaqus/Standard analysis.
ss_s_flsd.inp	Second Abaqus/Standard analysis.
sx_s_flsd.inp	Base problem for Abaqus/Standard to Abaqus/Explicit import.
sx_x_flsd.inp	Abaqus/Explicit analysis, imported from sx_s_flsd.inp.

MSFLD damage initiation (Material 5):

xs_x_msfld.inp	First Abaqus/Explicit analysis.
xs_s_msfld.inp	Abaqus/Standard analysis.
ss_s_msfld.inp	Second Abaqus/Standard analysis.
sx_s_msfld.inp	Base problem for Abaqus/Standard to Abaqus/Explicit import.
sx_x_msfld.inp	Abaqus/Explicit analysis, imported from sx_s_msfld.inp.

Damage evolution based on equivalent plastic displacement with linear softening (Material 6):

xs_x_ductile_displin.inp	First Abaqus/Explicit analysis.
xs_s_ductile_displin.inp	Abaqus/Standard analysis.

Damage evolution based on equivalent plastic displacement with tabular softening (Material 7):

xs_x_ductile_disptab.inp	First Abaqus/Explicit analysis.
xs_s_ductile_disptab.inp	Abaqus/Standard analysis.

Damage evolution based on equivalent plastic displacement with exponential softening (Material 8):

sx_s_ductile_dispexp.inp	First Abaqus/Standard analysis.
sx_x_ductile_dispexp.inp	Abaqus/Explicit analysis.
xs_s_ductile_dispexp.inp	Second Abaqus/Standard analysis.

Damage evolution based on energy dissipated during the damage process with linear softening (Material 9):

sx_s_ductile_enerlin.inp	First Abaqus/Standard analysis.
sx_x_ductile_enerlin.inp	Abaqus/Explicit analysis.
xs_s_ductile_enerlin.inp	Second Abaqus/Standard analysis.

Damage evolution based on energy dissipated during the damage process with exponential softening (Material 10):

xs_x_ductile_enerexp.inp	First Abaqus/Explicit analysis.
xs_s_ductile_enerexp.inp	Abaqus/Standard analysis.

Transferring results with hyperelasticity

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

CPS4R CPS3 CPE4R CPE3 CAX4R CAX3 C3D8R C3D4 C3D6
CPS6M CPE6M CAX6M C3D10M
M3D4R M3D3 S3R S4R SAX1

Problem description

The verification tests in this section consist of one-element models that are subjected to monotonically increasing tensile loads in a sequential import analysis. A slightly compressible hyperelastic material is used in the analyses. The sequence of tests involves transferring results from Abaqus/Standard to Abaqus/Explicit and then back into Abaqus/Standard.

Verification tests are also included for some first-order reduced-integration elements with enhanced hourglass control.

Four types of hyperelastic strain energy potentials are considered: the polynomial, Ogden, Arruda-Boyce, and van der Waals forms. For the tests using the polynomial strain energy potential, $N = 1$ and the material properties are

$C_{10} = 80$

$C_{01} = 20$

$D_1 = 0.001$

For the Ogden strain energy potential, $N = 2$ with

$\mu_1 = 160$

$\alpha_1 = 2$

$\mu_2 = 40$

$\alpha_2 = -2$

$D_1 = 0.001$

$D_2 = 0.00025$

For the Arruda-Boyce strain energy potential, the material properties are

$\mu = 200$

$\lambda_m = 5$

$D = 0.001$

For the van der Waals strain energy potential, the material properties are

$\mu = 200$

$\lambda_m = 10$

$a = 0.1$

$\beta = 0.0$

$D = 0.001$

Results and discussion

The results demonstrate that the hyperelasticity material model is transferred successfully between Abaqus/Explicit and Abaqus/Standard.

Input files

CPS4R element tests:

sx_s_cps4r_hyper.inp	First Abaqus/Standard analysis.
sx_x_cps4r_hyper.inp	Abaqus/Explicit analysis.
xs_s_cps4r_hyper.inp	Second Abaqus/Standard analysis.

CPS3 element tests:

sx_s_cps3_hyper.inp	First Abaqus/Standard analysis.
sx_x_cps3_hyper.inp	Abaqus/Explicit analysis.
xs_s_cps3_hyper.inp	Second Abaqus/Standard analysis.

CPE4R element tests:

sx_s_cpe4r_hyper.inp	First Abaqus/Standard analysis.
sx_x_cpe4r_hyper.inp	Abaqus/Explicit analysis.
xs_s_cpe4r_hyper.inp	Second Abaqus/Standard analysis.

CPE3 element tests:

sx_s_cpe3_hyper.inp	First Abaqus/Standard analysis.
sx_x_cpe3_hyper.inp	Abaqus/Explicit analysis.
xs_s_cpe3_hyper.inp	Second Abaqus/Standard analysis.

CAX4R element tests:

sx_s_cax4r_hyper.inp	First Abaqus/Standard analysis.
sx_s_cax4r_hyper_enhg.inp	First Abaqus/Standard analysis with enhanced hourglass control.
sx_x_cax4r_hyper.inp	Abaqus/Explicit analysis.
sx_x_cax4r_hyper_enhg.inp	Abaqus/Explicit analysis with enhanced hourglass control.
xs_s_cax4r_hyper.inp	Second Abaqus/Standard analysis.
xs_s_cax4r_hyper_enhg.inp	Second Abaqus/Standard analysis with enhanced hourglass control.

CAX3 element tests:

sx_s_cax3_hyper.inp	First Abaqus/Standard analysis.

sx_x_cax3_hyper.inp	Abaqus/Explicit analysis.
xs_s_cax3_hyper.inp	Second Abaqus/Standard analysis.

C3D8R element tests:

sx_s_c3d8r_hyper.inp	First Abaqus/Standard analysis.
sx_x_c3d8r_hyper.inp	Abaqus/Explicit analysis.
xs_s_c3d8r_hyper.inp	Second Abaqus/Standard analysis.

C3D4 element tests:

sx_s_c3d4_hyper.inp	First Abaqus/Standard analysis.
sx_x_c3d4_hyper.inp	Abaqus/Explicit analysis.
xs_s_c3d4_hyper.inp	Second Abaqus/Standard analysis.

C3D6 element tests:

sx_s_c3d6_hyper.inp	First Abaqus/Standard analysis.
sx_x_c3d6_hyper.inp	Abaqus/Explicit analysis.
xs_s_c3d6_hyper.inp	Second Abaqus/Standard analysis.

CPS6M element tests:

sx_s_cps6m_hyper.inp	First Abaqus/Standard analysis.
sx_x_cps6m_hyper.inp	Abaqus/Explicit analysis.
xs_s_cps6m_hyper.inp	Second Abaqus/Standard analysis.

CPE6M element tests:

sx_s_cpe6m_hyper.inp	First Abaqus/Standard analysis.
sx_x_cpe6m_hyper.inp	Abaqus/Explicit analysis.
xs_s_cpe6m_hyper.inp	Second Abaqus/Standard analysis.

CAX6M element tests:

sx_s_cax6m_hyper.inp	First Abaqus/Standard analysis.
sx_x_cax6m_hyper.inp	Abaqus/Explicit analysis.
xs_s_cax6m_hyper.inp	Second Abaqus/Standard analysis.

C3D10M element tests:

sx_s_c3d10m_hyper.inp	First Abaqus/Standard analysis.
sx_x_c3d10m_hyper.inp	Abaqus/Explicit analysis.
xs_s_c3d10m_hyper.inp	Second Abaqus/Standard analysis.

M3D4R element tests:

sx_s_m3d4r_hyper.inp	First Abaqus/Standard analysis.
sx_x_m3d4r_hyper.inp	Abaqus/Explicit analysis.
xs_s_m3d4r_hyper.inp	Second Abaqus/Standard analysis.

M3D3 element tests:

sx_s_m3d3_hyper.inp	First Abaqus/Standard analysis.
sx_x_m3d3_hyper.inp	Abaqus/Explicit analysis.
xs_s_m3d3_hyper.inp	Second Abaqus/Standard analysis.

S3R element tests:

sx_s_s3r_hyper.inp	First Abaqus/Standard analysis.
sx_x_s3r_hyper.inp	Abaqus/Explicit analysis.
xs_s_s3r_hyper.inp	Second Abaqus/Standard analysis.

S4R element tests:

sx_s_s4r_hyper.inp	First Abaqus/Standard analysis.
sx_s_s4r_hyper_enhg.inp	First Abaqus/Standard analysis with enhanced hourglass control.
sx_x_s4r_hyper.inp	Abaqus/Explicit analysis.
sx_x_s4r_hyper_enhg.inp	Abaqus/Explicit analysis with enhanced hourglass control.
xs_s_s4r_hyper.inp	Second Abaqus/Standard analysis.
xs_s_s4r_hyper_enhg.inp	Second Abaqus/Standard analysis with enhanced hourglass control.

SAX1 element tests:

sx_s_sax1_hyper.inp	First Abaqus/Standard analysis.
sx_x_sax1_hyper.inp	Abaqus/Explicit analysis.
xs_s_sax1_hyper.inp	Second Abaqus/Standard analysis.

Transferring results with viscoelasticity

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D8R CPE4R CPS4R CAX4R S4R M3D4R SAX1 B31 T3D2

Problem description

The verification tests in this section consist of one-element relaxation tests with viscoelastic materials and involve transferring results between Abaqus/Standard and Abaqus/Explicit, from one Abaqus/Standard analysis to another Abaqus/Standard analysis, or from one Abaqus/Explicit analysis to another Abaqus/Explicit analysis. Both small- and finite-strain time domain viscoelasticity with all possible stress states are tested. For finite-strain viscoelasticity the polynomial, Ogden, Marlow, Van der Waals, and Arruda-Boyce forms of strain energy potentials as well as hyperfoams are considered. The elements are loaded in tension or shear followed by relaxation at constant strain. The tests are performed for different combinations of the import capability. The results are transferred at the end of the first step of the original analysis, and relaxation is allowed to continue in the import analysis.

Results and discussion

The results demonstrate that the viscoelastic material model is transferred successfully between Abaqus analyses. The relaxation behavior of the material after import is consistent with the original analysis, where relaxation is allowed to continue in a second step for the same amount of time as the import analysis.

Input files

Transferring results from Abaqus/Standard to Abaqus/Explicit:

sx_s_visco_elas.inp	Abaqus/Standard analysis, viscoelasticity with the linear elastic material model.
sx_x_visco_elas_ny.inp	Abaqus/Explicit import analysis, UPDATE=NO, STATE=YES.
sx_x_visco_elas_yy.inp	Abaqus/Explicit import analysis, UPDATE=YES, STATE=YES.
sx_x_visco_elas_yn.inp	Abaqus/Explicit import analysis, UPDATE=YES, STATE=NO.
sx_s_visco_poly.inp	Abaqus/Standard analysis, viscoelasticity with the polynomial hyperelastic material model.
sx_x_visco_poly_ny.inp	Abaqus/Explicit import analysis, UPDATE=NO, STATE=YES.
sx_x_visco_poly_yy.inp	Abaqus/Explicit import analysis, UPDATE=YES, STATE=YES.
sx_x_visco_poly_yn.inp	Abaqus/Explicit import analysis, UPDATE=YES, STATE=NO.
sx_s_visco_ogden.inp	Abaqus/Standard analysis, viscoelasticity with the Ogden hyperelastic material model.
sx_x_visco_ogden_ny.inp	Abaqus/Explicit import analysis, UPDATE=NO, STATE=YES.
sx_x_visco_ogden_yy.inp	Abaqus/Explicit import analysis, UPDATE=YES, STATE=YES.
sx_x_visco_ogden_yn.inp	Abaqus/Explicit import analysis, UPDATE=YES, STATE=NO.

sx_s_marlow_visco.inp	Abaqus/Standard analysis, viscoelasticity with the Marlow hyperelastic material model.
sx_x_visco_marlow_ny.inp	Abaqus/Explicit import analysis, UPDATE=NO, STATE=YES.
sx_x_visco_marlow_yy.inp	Abaqus/Explicit import analysis, UPDATE=YES, STATE=YES.
sx_x_visco_marlow_yn.inp	Abaqus/Explicit import analysis, UPDATE=YES, STATE=NO.
sx_s_visco_vanderwaals.inp	Abaqus/Standard analysis, viscoelasticity with the Van der Waals hyperelastic material model.
sx_x_visco_vanderwaals_ny.inp	Abaqus/Explicit import analysis, UPDATE=NO, STATE=YES.
sx_x_visco_vanderwaals_yy.inp	Abaqus/Explicit import analysis, UPDATE=YES, STATE=YES.
sx_x_visco_vanderwaals_yn.inp	Abaqus/Explicit import analysis, UPDATE=YES, STATE=NO.
sx_s_visco_arruda.inp	Abaqus/Standard analysis, viscoelasticity with the Arruda-Boyce hyperelastic material model.
sx_x_visco_arruda_ny.inp	Abaqus/Explicit import analysis, UPDATE=NO, STATE=YES.
sx_x_visco_arruda_yy.inp	Abaqus/Explicit import analysis, UPDATE=YES, STATE=YES.
sx_x_visco_arruda_yn.inp	Abaqus/Explicit import analysis, UPDATE=YES, STATE=NO.
sx_s_visco_hyperfoam.inp	Abaqus/Standard analysis, viscoelasticity with the hyperfoam material model.
sx_x_visco_hyperfoam_ny.inp	Abaqus/Explicit import analysis, UPDATE=NO, STATE=YES.
sx_x_visco_hyperfoam_yy.inp	Abaqus/Explicit import analysis, UPDATE=YES, STATE=YES.
sx_x_visco_hyperfoam_yn.inp	Abaqus/Explicit import analysis, UPDATE=YES, STATE=NO.
sx_s_visco_prony6.inp	Abaqus/Standard analysis, viscoelasticity with six Prony series terms.
sx_x_visco_prony6_ny.inp	Abaqus/Explicit import analysis, UPDATE=NO, STATE=YES.
sx_x_visco_prony6_yy.inp	Abaqus/Explicit import analysis, UPDATE=YES, STATE=YES.
sx_x_visco_prony6_yn.inp	Abaqus/Explicit import analysis, UPDATE=YES, STATE=NO.

Transferring results from Abaqus/Explicit to Abaqus/Standard:

xs_x_visco_elas.inp	Abaqus/Explicit analysis, viscoelasticity with the linear elastic material model.
xs_s_visco_elas_ny.inp	Abaqus/Standard import analysis, UPDATE=NO, STATE=YES.
xs_s_visco_elas_yy.inp	Abaqus/Standard import analysis, UPDATE=YES, STATE=YES.
xs_s_visco_elas_yn.inp	Abaqus/Standard import analysis, UPDATE=YES, STATE=NO.
xs_x_visco_poly.inp	Abaqus/Explicit analysis, viscoelasticity with the polynomial hyperelastic material model.
xs_s_visco_poly_ny.inp	Abaqus/Standard import analysis, UPDATE=NO, STATE=YES.
xs_s_visco_poly_yy.inp	Abaqus/Standard import analysis, UPDATE=YES, STATE=YES.
xs_s_visco_poly_yn.inp	Abaqus/Standard import analysis, UPDATE=YES, STATE=NO.

Transferring results from one Abaqus/Standard analysis to another Abaqus/Standard analysis:

ss_visco_elas.inp	Abaqus/Standard analysis, viscoelasticity with the linear elastic material model.
ss_i_visco_elas_ny.inp	Abaqus/Standard import analysis, UPDATE=NO, STATE=YES.
ss_i_visco_elas_yy.inp	Abaqus/Standard import analysis, UPDATE=YES, STATE=YES.
ss_i_visco_elas_yn.inp	Abaqus/Standard import analysis, UPDATE=YES, STATE=NO.
ss_visco_poly.inp	Abaqus/Explicit analysis, viscoelasticity with the polynomial hyperelastic material model.

ss_i_visco_poly_ny.inp	Abaqus/Standard import analysis, UPDATE=NO, STATE=YES.
ss_i_visco_poly_yy.inp	Abaqus/Standard import analysis, UPDATE=YES, STATE=YES.
ss_i_visco_poly_yn.inp	Abaqus/Standard import analysis, UPDATE=YES, STATE=NO.

Transferring results from one Abaqus/Explicit analysis to another Abaqus/Explicit analysis:

xx1_visco_aniso.inp	Abaqus/Explicit analysis, viscoelasticity with the linear elastic anisotropic material.
xx2_visco_aniso_ny.inp	Abaqus/Explicit import analysis, UPDATE=NO, STATE=YES.
xx1_visco_ortho.inp	Abaqus/Explicit analysis, viscoelasticity with the linear elastic orthotropic material.
xx2_visco_ortho_ny.inp	Abaqus/Explicit import analysis, UPDATE=NO, STATE=YES.

Transferring results for a hyperelastic sheet with a circular hole

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

CPS4R

Problem description

This test considers the uniform large stretching of a thin, initially square sheet containing a centrally located circular hole. The sheet is subjected to monotonically increasing loads as the analysis is carried out, first with Abaqus/Standard, then with Abaqus/Explicit, and finally with Abaqus/Standard again. The test is an additional demonstration of the use of the import capability when hyperelastic materials are used in the analysis.

The undeformed sheet is 2 mm (0.079 in) thick and 165 mm (6.5 in) on each side. It has a centrally located internal hole of radius 6.35 mm (0.25 in). CPS4R elements are used in the finite element model of the sheet. Plane stress conditions are assumed. The sheet is stretched in the x-direction while it is constrained from stretching in the y-direction. Symmetry conditions allow only a quarter of the sheet to be modeled.

A polynomial hyperelasticity model is used to describe the material behavior. The model is assumed to be slightly compressible since Abaqus/Explicit does not allow incompressible material behavior. Thus, the constants D_i are set to small values. The material parameters used in the analysis are:

$C_{10} = 27.02$

$C_{01} = 1.42$

$C_{20} = -0.27$

$C_{11} = 0.0$

$C_{02} = 0.0$

$C_{30} = 0.00654$

$C_{21} = 0.0$

$C_{12} = 0.0$

$C_{03} = 0.0$

$D_1 = 0.001$

$D_2 = 0.001$

$D_3 = 0.001$

The testing of the results transfer capability consists of three separate analyses. A static analysis is conducted in Abaqus/Standard, wherein the sheet is stretched to a width of 520.7 mm (20.5 in). Subsequent quasi-static stretching of the sheet by an additional amount of 55.6 mm (46.5 in) is analyzed in Abaqus/Explicit. The final

phase of stretching to a total value of 1181 mm (46.5 in) is analyzed in an another static procedure in Abaqus/Standard.

Results and discussion

The final results of the analysis using the results transfer capability agree well with the results of an analysis conducted entirely within Abaqus/Standard.

Input files

sx_s_holehyper.inp	First Abaqus/Standard analysis.
sx_x_holehyper.inp	Abaqus/Explicit analysis.
xs_s_holehyper.inp	Second Abaqus/Standard analysis.

Transferring results with hyperfoam

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

CPS4R CPS3 CPE4R CPE3 CAX4R CAX3 C3D8R C3D4 C3D6
M3D4R M3D3 S3R S4R SAX1
C3D10M CPE6M CPS6M CAX6M

Problem description

The verification tests in this section consist of one-element models that are subjected to monotonically increasing compressive loads in a sequential import analysis. The sequence of tests involves transferring results from Abaqus/Standard to Abaqus/Explicit and then back into Abaqus/Standard.

Verification tests are also included for some first-order reduced-integration elements with enhanced hourglass control.

The material properties are the same as those used in *Fitting of elastomeric foam test data*.

Results and discussion

The results demonstrate that the hyperfoam material model is transferred successfully between Abaqus/Explicit and Abaqus/Standard.

Input files

CPS4R element tests:

sx_s_cps4r_hfoam.inp	First Abaqus/Standard analysis.
sx_s_cps4r_hfoam_enhg.inp	First Abaqus/Standard analysis with enhanced hourglass control.
sx_x_cps4r_hfoam.inp	Abaqus/Explicit analysis.
sx_x_cps4r_hfoam_enhg.inp	Abaqus/Explicit analysis with enhanced hourglass control.
xs_s_cps4r_hfoam.inp	Second Abaqus/Standard analysis.
xs_s_cps4r_hfoam_enhg.inp	Second Abaqus/Standard analysis with enhanced hourglass control.

CPS3 element tests:

sx_s_cps3_hfoam.inp	First Abaqus/Standard analysis.
sx_x_cps3_hfoam.inp	Abaqus/Explicit analysis.
xs_s_cps3_hfoam.inp	Second Abaqus/Standard analysis.

CPE4R element tests:

sx_s_cpe4r_hfoam.inp	First Abaqus/Standard analysis.
sx_x_cpe4r_hfoam.inp	Abaqus/Explicit analysis.

xs_s_cpe4r_hfoam.inp	Second Abaqus/Standard analysis.
CPE3 element tests:	
sx_s_cpe3_hfoam.inp	First Abaqus/Standard analysis.
sx_x_cpe3_hfoam.inp	Abaqus/Explicit analysis.
xs_s_cpe3_hfoam.inp	Second Abaqus/Standard analysis.
CAX4R element tests:	
sx_s_cax4r_hfoam.inp	First Abaqus/Standard analysis.
sx_x_cax4r_hfoam.inp	Abaqus/Explicit analysis.
xs_s_cax4r_hfoam.inp	Second Abaqus/Standard analysis.
CAX3 element tests:	
sx_s_cax3_hfoam.inp	First Abaqus/Standard analysis.
sx_x_cax3_hfoam.inp	Abaqus/Explicit analysis.
xs_s_cax3_hfoam.inp	Second Abaqus/Standard analysis.
C3D8R element tests:	
sx_s_c3d8r_hfoam.inp	First Abaqus/Standard analysis.
sx_s_c3d8r_hfoam_enhg.inp	First Abaqus/Standard analysis with enhanced hourglass control.
sx_x_c3d8r_hfoam.inp	Abaqus/Explicit analysis.
sx_x_c3d8r_hfoam_enhg.inp	Abaqus/Explicit analysis with enhanced hourglass control.
xs_s_c3d8r_hfoam.inp	Second Abaqus/Standard analysis.
xs_s_c3d8r_hfoam_enhg.inp	Second Abaqus/Standard analysis with enhanced hourglass control.
C3D4 element tests:	
sx_s_c3d4_hfoam.inp	First Abaqus/Standard analysis.
sx_x_c3d4_hfoam.inp	Abaqus/Explicit analysis.
xs_s_c3d4_hfoam.inp	Second Abaqus/Standard analysis.
C3D6 element tests:	
sx_s_c3d6_hfoam.inp	First Abaqus/Standard analysis.
sx_x_c3d6_hfoam.inp	Abaqus/Explicit analysis.
xs_s_c3d6_hfoam.inp	Second Abaqus/Standard analysis.
C3D10M element tests:	
sx_s_c3d10m_hfoam.inp	First Abaqus/Standard analysis.
sx_x_c3d10m_hfoam.inp	Abaqus/Explicit analysis.
xs_s_c3d10m_hfoam.inp	Second Abaqus/Standard analysis.
CPE6M element tests:	
sx_s_cpe6m_hfoam.inp	First Abaqus/Standard analysis.
sx_x_cpe6m_hfoam.inp	Abaqus/Explicit analysis.
xs_s_cpe6m_hfoam.inp	Second Abaqus/Standard analysis.
CPS6M element tests:	
sx_s_cps6m_hfoam.inp	First Abaqus/Standard analysis.

sx_x_cps6m_hfoam.inp	Abaqus/Explicit analysis.
xs_s_cps6m_hfoam.inp	Second Abaqus/Standard analysis.

CAX6M element tests:

sx_s_cax6m_hfoam.inp	First Abaqus/Standard analysis.
sx_x_cax6m_hfoam.inp	Abaqus/Explicit analysis.
xs_s_cax6m_hfoam.inp	Second Abaqus/Standard analysis.

M3D4R element tests:

sx_s_m3d4r_hfoam.inp	First Abaqus/Standard analysis.
sx_s_m3d4r_hfoam_enhg.inp	First Abaqus/Standard analysis with enhanced hourglass control.
sx_x_m3d4r_hfoam.inp	Abaqus/Explicit analysis.
sx_x_m3d4r_hfoam_enhg.inp	Abaqus/Explicit analysis with enhanced hourglass control.
xs_s_m3d4r_hfoam.inp	Second Abaqus/Standard analysis.
xs_s_m3d4r_hfoam_enhg.inp	Second Abaqus/Standard analysis with enhanced hourglass control.

M3D3 element tests:

sx_s_m3d3_hfoam.inp	First Abaqus/Standard analysis.
sx_x_m3d3_hfoam.inp	Abaqus/Explicit analysis.
xs_s_m3d3_hfoam.inp	Second Abaqus/Standard analysis.

S3R element tests:

sx_s_s3r_hfoam.inp	First Abaqus/Standard analysis.
sx_x_s3r_hfoam.inp	Abaqus/Explicit analysis.
xs_s_s3r_hfoam.inp	Second Abaqus/Standard analysis.

S4R element tests:

sx_s_s4r_hfoam.inp	First Abaqus/Standard analysis.
sx_x_s4r_hfoam.inp	Abaqus/Explicit analysis.
xs_s_s4r_hfoam.inp	Second Abaqus/Standard analysis.

SAX1 element tests:

sx_s_sax1_hfoam.inp	First Abaqus/Standard analysis.
sx_x_sax1_hfoam.inp	Abaqus/Explicit analysis.
xs_s_sax1_hfoam.inp	Second Abaqus/Standard analysis.

Transferring results with orientation

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Transferring results between Abaqus/Explicit and Abaqus/Standard

Elements tested

C3D8R CPE4R CPS4R C3D10M CPE6M CPS6M M3D4R S4R S4

Problem description

The verification tests in this section consist of testing the transfer of the orientation definitions between Abaqus/Standard and Abaqus/Explicit and vice versa. The tests involve single elements in simple shear subjected to monotonically increasing loads.

Verification tests are also included for some first-order reduced-integration elements with enhanced hourglass control.

The material model used for all the tests is orthotropic elasticity defined by specifying the generalized Young's moduli, the Poisson's ratios, and the shear moduli in the principal directions. The following material properties are used (the units are not important):

$E_1 = 200 \times 10^9$

$E_2 = 100 \times 10^9$

$E_3 = 100 \times 10^9$

$\nu_{12} = 0.3$

$\nu_{13} = 0.23$

$\nu_{23} = 0.34$

$G_{12} = 76.9 \times 10^9$

$G_{13} = 76.9 \times 10^9$

$G_{23} = 9.0 \times 10^9$

Density = 7850.

Since nonisotropic material behavior is defined, a local coordinate system is necessary for the anisotropic behavior to be associated with the material directions. Nondefault orientations are specified in the original analysis so that the local material directions are inclined at 45° to the element local directions. A large-displacement analysis is used, which results in the nondefault local coordinate system rotating with the average rigid body motion at the material point. The orientation definitions are transferred to the import analysis by default.

The resulting stresses, strains, section forces, and section strains, wherever applicable, are all reported in the local coordinate system by default.

A verification test is also carried out for a composite shell with three layers. An S4R element is used. A section orientation of 45° is defined with respect to the local directions. Additional orientations of 15°, 30°, and 45° with respect to the newly defined section orientation are defined for material calculations for individual layers. The material model used for the tests is orthotropic elasticity, which is defined above.

Results and discussion

The results demonstrate that orientation definitions are transferred successfully between Abaqus/Explicit and Abaqus/Standard.

Input files

Transfer from Abaqus/Standard to Abaqus/Explicit

C3D8R element tests:

sx_s_c3d8r_or.inp	Abaqus/Standard analysis.
sx_x_c3d8r_or_n_y.inp	Abaqus/Explicit analysis with UPDATE=NO and STATE=YES.
sx_x_c3d8r_or_y_y.inp	Abaqus/Explicit analysis with UPDATE=YES and STATE=YES.

Tests of three C3D8R elements:

sx_s_c3d8r_3or.inp	Abaqus/Standard analysis.
sx_x_c3d8r_3or_n_y.inp	Abaqus/Explicit analysis with UPDATE=NO and STATE=YES.

CPE4R element tests:

sx_s_cpe4r_or.inp	Abaqus/Standard analysis.
sx_s_cpe4r_or_enhg.inp	Abaqus/Standard analysis with enhanced hourglass control.
sx_x_cpe4r_or_n_y.inp	Abaqus/Explicit analysis with UPDATE=NO and STATE=YES.
sx_x_cpe4r_or_n_y_enhg.inp	Abaqus/Explicit analysis with UPDATE=NO, STATE=YES and enhanced hourglass control.
sx_x_cpe4r_or_y_y.inp	Abaqus/Explicit analysis with UPDATE=YES and STATE=YES.
sx_x_cpe4r_or_y_y_enhg.inp	Abaqus/Explicit analysis with UPDATE=YES, STATE=YES and enhanced hourglass control.

CPS4R element tests:

sx_s_cps4r_or.inp	Abaqus/Standard analysis.
sx_x_cps4r_or_n_y.inp	Abaqus/Explicit analysis with UPDATE=NO and STATE=YES.
sx_x_cps4r_or_y_y.inp	Abaqus/Explicit analysis with UPDATE=YES and STATE=YES.

C3D10M element tests:

sx_s_c3d10m_or.inp	Abaqus/Standard analysis.
sx_x_c3d10m_or_n_y.inp	Abaqus/Explicit analysis with UPDATE=NO and STATE=YES.
sx_x_c3d10m_or_y_y.inp	Abaqus/Explicit analysis with UPDATE=YES and STATE=YES.

CPE6M element tests:

sx_s_cpe6m_or.inp	Abaqus/Standard analysis.
sx_x_cpe6m_or_n_y.inp	Abaqus/Explicit analysis with UPDATE=NO and STATE=YES.
sx_x_cpe6m_or_y_y.inp	Abaqus/Explicit analysis with UPDATE=YES and STATE=YES.

CPS6M element tests:

sx_s_cps6m_or.inp	Abaqus/Standard analysis.

sx_x_cps6m_or_n_y.inp	Abaqus/Explicit analysis with UPDATE=NO and STATE=YES.
sx_x_cps6m_or_y_y.inp	Abaqus/Explicit analysis with UPDATE=YES and STATE=YES.

M3D4R element tests:

sx_s_m3d4r_or.inp	Abaqus/Standard analysis.
sx_s_m3d4r_or_enhg.inp	Abaqus/Standard analysis with enhanced hourglass control.
sx_x_m3d4r_or_n_y.inp	Abaqus/Explicit analysis with UPDATE=NO and STATE=YES.
sx_x_m3d4r_or_n_y_enhg.inp	Abaqus/Explicit analysis with UPDATE=NO, STATE=YES and enhanced hourglass control.
sx_x_m3d4r_or_y_y.inp	Abaqus/Explicit analysis with UPDATE=YES and STATE=YES.
sx_x_m3d4r_or_y_y_enhg.inp	Abaqus/Explicit analysis with UPDATE=YES, STATE=YES and enhanced hourglass control.

S4R element tests:

sx_s_s4r_or.inp	Abaqus/Standard analysis.
sx_x_s4r_or_n_y.inp	Abaqus/Explicit analysis with UPDATE=NO and STATE=YES.
sx_x_s4r_or_y_y.inp	Abaqus/Explicit analysis with UPDATE=YES and STATE=YES.

Composite shell tests:

sx_s_s4r_com_or.inp	Abaqus/Standard analysis.
sx_s_s4r_com_or_enhg.inp	Abaqus/Standard analysis with enhanced hourglass control.
sx_x_s4r_com_or_n_y.inp	Abaqus/Explicit analysis with UPDATE=NO and STATE=YES.
sx_x_s4r_com_or_n_y_enhg.inp	Abaqus/Explicit analysis with UPDATE=NO, STATE=YES and enhanced hourglass control.
sx_x_s4r_com_or_y_y.inp	Abaqus/Explicit analysis with UPDATE=YES and STATE=YES.
sx_x_s4r_com_or_y_y_enhg.inp	Abaqus/Explicit analysis with UPDATE=YES, STATE=YES and enhanced hourglass control.

Transfer from Abaqus/Explicit to Abaqus/Standard

C3D8R element tests:

xs_x_c3d8r_or.inp	Abaqus/Explicit analysis.
xs_s_c3d8r_or_n_y.inp	Abaqus/Standard analysis with UPDATE=NO and STATE=YES.
xs_s_c3d8r_or_y_y.inp	Abaqus/Standard analysis with UPDATE=YES and STATE=YES.

CPE4R element tests:

xs_x_cpe4r_or.inp	Abaqus/Explicit analysis.
xs_s_cpe4r_or_n_y.inp	Abaqus/Standard analysis with UPDATE=NO and STATE=YES.
xs_s_cpe4r_or_y_y.inp	Abaqus/Standard analysis with UPDATE=YES and STATE=YES.

CPS4R element tests:

xs_x_cps4r_or.inp	Abaqus/Explicit analysis.
xs_s_cps4r_or_n_y.inp	Abaqus/Standard analysis with UPDATE=NO and STATE=YES.
xs_s_cps4r_or_y_y.inp	Abaqus/Standard analysis with UPDATE=YES and STATE=YES.

C3D10M element tests:

xs_x_c3d10m_or.inp	Abaqus/Explicit analysis.
xs_s_c3d10m_or_n_y.inp	Abaqus/Standard analysis with UPDATE=NO and STATE=YES.

xs_s_c3d10m_or_y_y.inp	Abaqus/Standard analysis with UPDATE=YES and STATE=YES.

CPE6M element tests:

xs_x_cpe6m_or.inp	Abaqus/Explicit analysis.
xs_s_cpe6m_or_n_y.inp	Abaqus/Standard analysis with UPDATE=NO and STATE=YES.
xs_s_cpe6m_or_y_y.inp	Abaqus/Standard analysis with UPDATE=YES and STATE=YES.

CPS6M element tests:

xs_x_cps6m_or.inp	Abaqus/Explicit analysis.
xs_s_cps6m_or_n_y.inp	Abaqus/Standard analysis with UPDATE=NO and STATE=YES.
xs_s_cps6m_or_y_y.inp	Abaqus/Standard analysis with UPDATE=YES and STATE=YES.

M3D4R element tests:

xs_x_m3d4r_or.inp	Abaqus/Explicit analysis.
xs_s_m3d4r_or_n_y.inp	Abaqus/Standard analysis with UPDATE=NO and STATE=YES.
xs_s_m3d4r_or_y_y.inp	Abaqus/Standard analysis with UPDATE=YES and STATE=YES.

S4 element tests:

xs_x_s4_or.inp	Abaqus/Explicit analysis.
xs_s_s4_or_n_y.inp	Abaqus/Standard analysis with UPDATE=NO and STATE=YES.
xs_s_s4_or_y_y.inp	Abaqus/Standard analysis with UPDATE=YES and STATE=YES.

S4R element tests:

xs_x_s4r_or.inp	Abaqus/Explicit analysis.
xs_s_s4r_or_n_y.inp	Abaqus/Standard analysis with UPDATE=NO and STATE=YES.
xs_s_s4r_or_y_y.inp	Abaqus/Standard analysis with UPDATE=YES and STATE=YES.

Composite shell tests:

xs_x_s4r_com_or.inp	Abaqus/Explicit analysis.
xs_s_s4r_com_or_n_y.inp	Abaqus/Standard analysis with UPDATE=NO and STATE=YES.
xs_s_s4r_com_or_y_y.inp	Abaqus/Standard analysis with UPDATE=YES and STATE=YES.

Transferring results from one Abaqus/Standard analysis to another Abaqus/Standard analysis

Elements tested

C3D8 C3D8R CPE4 CPE4R CPS4 CPS4R C3D10M CPE6M CPS6M M3D4R S4R

Problem description

The verification tests in this section test the transfer of the orientation definitions from one Abaqus/Standard analysis to another. The tests involve single elements in simple shear subjected to monotonically increasing loads. The first analysis consists of two steps in which the element is subjected to simple shear loads. The second analysis imports the results from the end of the first step of the first analysis and subjects the element to the same loading as in the second step of the first analysis. The transfer of orientation is verified using the current material state import option with and without the reference configuration update option of the import feature in the second analysis.

The material model used for all the tests is the same as the one used in the previous section.

Since nonisotropic material behavior is defined, the local coordinate system is necessary for the anisotropic behavior to be associated with the material directions. Nondefault orientations are specified in the original analysis so that the local material directions are inclined at 45° to the element local directions. A large-displacement analysis is used, which results in the nondefault local coordinate system rotating with the average rigid body motion at the material point. The orientation definitions are transferred to the import analysis by default.

The resulting stresses, strains, section forces, and section strains, wherever applicable, are all reported in the local coordinate system by default.

A verification test is also carried out for a composite shell with three layers. An S4R element is used. A section orientation of 45° is defined with respect to the local directions. Additional orientations of 15°, 30°, and 45° with respect to the newly defined section orientation are defined for material calculations for individual layers. The material model used for the tests is orthotropic elasticity, as defined in the previous section.

Verification tests are also included for some first-order reduced-integration elements with enhanced hourglass control.

Results and discussion

The results from the two Abaqus/Standard analyses are identical when the material state is imported and the reference configuration is not updated. The stresses and material orientations are identical when the material state is imported and the reference configuration is updated; the strains differ because the reference configuration is updated.

Input files

C3D8 element tests:

ss1_c3d8_or.inp	First Abaqus/Standard analysis.
ss2_c3d8_or_n_y.inp	Abaqus/Standard **IMPORT* analysis, UPDATE=NO and STATE=YES.
ss2_c3d8_or_y_y.inp	Abaqus/Standard **IMPORT* analysis, UPDATE=YES and STATE=YES.

C3D8R element tests:

ss1_c3d8r_or.inp	First Abaqus/Standard analysis.
ss1_c3d8r_or_enhg.inp	First Abaqus/Standard analysis with enhanced hourglass control.
ss2_c3d8r_or_n_y.inp	Abaqus/Standard **IMPORT* analysis, UPDATE=NO and STATE=YES.
ss2_c3d8r_or_n_y_enhg.inp	Abaqus/Standard **IMPORT* analysis, UPDATE=NO and STATE=YES with enhanced hourglass control.
ss2_c3d8r_or_y_y.inp	Abaqus/Standard **IMPORT* analysis, UPDATE=YES and STATE=YES.
ss2_c3d8r_or_y_y_enhg.inp	Abaqus/Standard **IMPORT* analysis, UPDATE=YES and STATE=YES with enhanced hourglass control.

CPE4 element tests:

ss1_cpe4_or.inp	First Abaqus/Standard analysis.
ss2_cpe4_or_n_y.inp	Abaqus/Standard **IMPORT* analysis, UPDATE=NO and STATE=YES.
ss2_cpe4_or_y_y.inp	Abaqus/Standard **IMPORT* analysis, UPDATE=YES and STATE=YES.

CPE4R element tests:

ss1_cpe4r_or.inp	First Abaqus/Standard analysis.
ss1_cpe4r_or_enhg.inp	First Abaqus/Standard analysis with enhanced hourglass control.
ss2_cpe4r_or_n_y.inp	Abaqus/Standard **IMPORT* analysis, UPDATE=NO and STATE=YES.

ss2_cpe4r_or_n_y_enhg.inp	Abaqus/Standard **IMPORT* analysis, UPDATE=NO and STATE=YES with enhanced hourglass control.
ss2_cpe4r_or_y_y.inp	Abaqus/Standard **IMPORT* analysis, UPDATE=YES and STATE=YES.
ss2_cpe4r_or_y_y_enhg.inp	Abaqus/Standard **IMPORT* analysis, UPDATE=YES and STATE=YES with enhanced hourglass control.

CPS4 element tests:

ss1_cps4_or.inp	First Abaqus/Standard analysis.
ss2_cps4_or_n_y.inp	Abaqus/Standard **IMPORT* analysis, UPDATE=NO and STATE=YES.
ss2_cps4_or_y_y.inp	Abaqus/Standard **IMPORT* analysis, UPDATE=YES and STATE=YES.

CPS4R element tests:

ss1_cps4r_or.inp	First Abaqus/Standard analysis.
ss2_cps4r_or_n_y.inp	Abaqus/Standard **IMPORT* analysis, UPDATE=NO and STATE=YES.
ss2_cps4r_or_y_y.inp	Abaqus/Standard **IMPORT* analysis, UPDATE=YES and STATE=YES.

C3D10M element tests:

ss1_c3d10m_or.inp	First Abaqus/Standard analysis.
ss2_c3d10m_or_n_y.inp	Abaqus/Standard **IMPORT* analysis, UPDATE=NO and STATE=YES.
ss2_c3d10m_or_y_y.inp	Abaqus/Standard **IMPORT* analysis, UPDATE=YES and STATE=YES.

CPE6M element tests:

ss1_cpe6m_or.inp	First Abaqus/Standard analysis.
ss2_cpe6m_or_n_y.inp	Abaqus/Standard **IMPORT* analysis, UPDATE=NO and STATE=YES.
ss2_cpe6m_or_y_y.inp	Abaqus/Standard **IMPORT* analysis, UPDATE=YES and STATE=YES.

CPS6M element tests:

ss1_cps6m_or.inp	First Abaqus/Standard analysis.
ss2_cps6m_or_n_y.inp	Abaqus/Standard **IMPORT* analysis, UPDATE=NO and STATE=YES.
ss2_cps6m_or_y_y.inp	Abaqus/Standard **IMPORT* analysis, UPDATE=YES and STATE=YES.

M3D4R element tests:

ss1_m3d4r_or.inp	First Abaqus/Standard analysis.
ss1_m3d4r_or_enhg.inp	First Abaqus/Standard analysis with enhanced hourglass control.
ss2_m3d4r_or_n_y.inp	Abaqus/Standard **IMPORT* analysis, UPDATE=NO and STATE=YES.
ss2_m3d4r_or_n_y_enhg.inp	Abaqus/Standard **IMPORT* analysis, UPDATE=NO and STATE=YES with enhanced hourglass control.
ss2_m3d4r_or_y_y.inp	Abaqus/Standard **IMPORT* analysis, UPDATE=YES and STATE=YES.
ss2_m3d4r_or_y_y_enhg.inp	Abaqus/Standard **IMPORT* analysis, UPDATE=YES and STATE=YES with enhanced hourglass control.

S4R element tests:

ss1_s4r_or.inp	First Abaqus/Standard analysis.
ss2_s4r_or_n_y.inp	Abaqus/Standard **IMPORT* analysis, UPDATE=NO and STATE=YES.
ss2_s4r_or_y_y.inp	Abaqus/Standard **IMPORT* analysis, UPDATE=YES and STATE=YES.

Composite shell tests:

ss1_s4r_com_or.inp	First Abaqus/Standard analysis.
ss1_s4r_com_or_enhg.inp	First Abaqus/Standard analysis with enhanced hourglass control.
ss2_s4r_com_or_n_y.inp	Abaqus/Standard **IMPORT* analysis, UPDATE=NO and STATE=YES.
ss2_s4r_com_or_n_y_enhg.inp	Abaqus/Standard **IMPORT* analysis, UPDATE=NO and STATE=YES with enhanced hourglass control.
ss2_s4r_com_or_y_y.inp	Abaqus/Standard **IMPORT* analysis, UPDATE=YES and STATE=YES.
ss2_s4r_com_or_y_y_enhg.inp	Abaqus/Standard **IMPORT* analysis, UPDATE=YES and STATE=YES with enhanced hourglass control.

Miscellaneous results transfer tests

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

This section gives a brief description of tests that are conducted to verify the use of different options in Abaqus/Standard and Abaqus/Explicit.

Model change

Elements tested

CPE4R

Problem description

This test verifies that elements that are rendered inactive in Abaqus/Standard are not imported into Abaqus/Explicit. The finite element model consists of three CPE4R elements. The analysis in Abaqus/Standard consists of four steps. In the first step the model is subjected to a tensile load, in Step 2 two of the elements are rendered inactive, in Step 3 one of these elements is reactivated, and finally in Step 4 the two active elements are subjected to an increased tensile load. The results from the end of Step 3 of the Abaqus/Standard analysis are imported into Abaqus/Explicit. Only the two active elements are imported; these two elements are then subjected to the same tensile loads as in Step 4 of the Abaqus/Standard analysis. This test is conducted with CPE4R elements. The material definition and loading are not important.

Results and discussion

The results at the end of the Abaqus/Explicit import analysis are identical to the results at the end of the Abaqus/Standard analysis. The results demonstrate that the effects of using the element removal capability are transferred correctly between Abaqus/Explicit and Abaqus/Standard. In addition, the results demonstrate that elements that are inactive in an Abaqus/Standard analysis will not be imported into Abaqus/Explicit.

Input files

sx_s_cpe4r_mc.inp	Abaqus/Standard analysis.
sx_x_cpe4r_mc.inp	Abaqus/Explicit analysis.

Frequency analysis after import

Elements tested

CPE4R C3D8R M3D4R S4R

Problem description

The following set of tests involves importing the results from Abaqus/Explicit and then conducting a frequency analysis in Abaqus/Standard. The model consists of a single element subjected to tensile load. Linear isotropic elasticity is used to describe the material behavior.

Verification tests of the enhanced hourglass control method are also included.

Results and discussion

The results demonstrate that frequency definitions are transferred correctly between Abaqus/Explicit and Abaqus/Standard.

Input files

CPE4R element tests:

xs_x_cpe4r_t.inp	Abaqus/Explicit analysis.
xs_s_cpe4r_fr_y_y.inp	Abaqus/Standard analysis.

C3D8R element tests:

xs_x_c3d8r_t.inp	Abaqus/Explicit analysis.
xs_s_c3d8r_fr_y_y.inp	Abaqus/Standard analysis.

M3D4R element tests:

xs_x_m3d4r_t.inp	Abaqus/Explicit analysis.
xs_x_m3d4r_t_enhg.inp	Abaqus/Explicit analysis with enhanced hourglass control.
xs_s_m3d4r_fr_y_y.inp	Abaqus/Standard analysis.
xs_s_m3d4r_fr_y_y_enhg.inp	Abaqus/Standard analysis with enhanced hourglass control.

S4R element tests:

xs_x_s4r_t.inp	Abaqus/Explicit analysis.
xs_s_s4r_fr_y_y.inp	Abaqus/Standard analysis.

Displacement formulation in Abaqus/Explicit and Abaqus/Standard

Elements tested

CPE4R

Problem description

These tests involve examples considering geometric nonlinearities. If geometric nonlinearities are considered in the original analysis, they will be considered by default in the subsequent import analysis as well, and the setting for geometric nonlinearities in the second analysis cannot be changed. If the geometric nonlinearities are not considered in the original analysis, they will also be neglected in the first step of the import analysis, and the reference configuration will not be updated. In this case, the settings can be changed if required.

The test consists of a single element subjected to monotonically increasing tensile loads. The small-displacement formulation is used in the Abaqus/Explicit analysis. The results are then imported into Abaqus/Standard. Two tests are carried out in Abaqus/Standard, one with the large-displacement formulation and one with the small-displacement formulation. Linear isotropic elastic properties for the material are assumed.

A similar test is conducted when the transfer is from Abaqus/Standard into Abaqus/Explicit.

Verification tests of the enhanced hourglass control method are also included.

Results and discussion

The results demonstrate that the value of the displacement formulation is transferred correctly between Abaqus/Explicit and Abaqus/Standard.

Input files

Transfer from Abaqus/Standard to Abaqus/Explicit

sx_s_cpe4r_nlg.inp	Abaqus/Standard analysis.
sx_s_cpe4r_nlg_enhg.inp	Abaqus/Standard analysis with enhanced hourglass control.
sx_x_cpe4r_nlg_n.inp	Abaqus/Explicit analysis with NLGEOM=NO.
sx_x_cpe4r_nlg_n_enhg.inp	Abaqus/Explicit analysis with NLGEOM=NO and enhanced hourglass control.
sx_x_cpe4r_nlg_y.inp	Abaqus/Explicit analysis with NLGEOM=YES.
sx_x_cpe4r_nlg_y_enhg.inp	Abaqus/Explicit analysis with NLGEOM=YES and enhanced hourglass control.

Transfer from Abaqus/Explicit to Abaqus/Standard

xs_x_cpe4r_nlg.inp	Abaqus/Explicit analysis.
xs_s_cpe4r_nlg_n.inp	Abaqus/Standard analysis with NLGEOM=NO.
xs_s_cpe4r_nlg_y.inp	Abaqus/Standard analysis with NLGEOM=YES.

Initial stresses and equivalent plastic strains

Elements tested

CPE4R

Problem description

The following tests verify the application of initial stresses and equivalent plastic strains in an import analysis. Initial stresses and equivalent plastic strains can be specified in an import analysis only when the material state is not imported.

A sequential analysis consisting of transfer from Abaqus/Explicit to Abaqus/Standard and then back to Abaqus/Explicit is conducted. The model consists of a single CPE4R element subjected to tensile loads. The material state is not imported, and the material behavior is described by linear isotropic elasticity with Mises plasticity. In the Abaqus/Standard analysis both initial equivalent plastic strains and initial stresses are prescribed, while in the second Abaqus/Explicit analysis only the stresses are prescribed.

The following material properties are used (the units are not important):

Elasticity

Young's modulus, E=200.0×10^9

Poisson's ratio, ν=0.3

Plasticity (Hardening)

Yield stress	Plastic strain
200.0E7	0.0000
220.0E7	0.001
240.0E7	0.01

Results and discussion

The results demonstrate that initial stresses and equivalent plastic strains are transferred correctly between Abaqus/Explicit and Abaqus/Standard.

Input files

xs_x_cpe4r_t.inp	First Abaqus/Explicit analysis.
xs_s_cpe4r_in_s.inp	Abaqus/Standard analysis.
sx_x_cpe4r_in_s.inp	Second Abaqus/Explicit analysis.

Initial angular velocities

Elements tested

S4R

Problem description

The application of initial velocities in terms of an angular velocity in an import analysis is tested. The transfer of results is from Abaqus/Standard into Abaqus/Explicit. The analysis in Abaqus/Standard involves subjecting a single S4R element to a centrifugal force. A static procedure is used in Abaqus/Standard for this purpose. The velocities are zero since the Abaqus/Standard analysis is a static analysis. Initial angular velocities are prescribed on the nodes of the imported element in Abaqus/Explicit to allow the spinning of the element about a particular axis. Linear isotropic elasticity is used to describe the material behavior.

Results and discussion

The results demonstrate that initial angular velocities are transferred correctly between Abaqus/Explicit and Abaqus/Standard.

Input files

sx_s_s4r_rv.inp	Abaqus/Standard analysis.
sx_x_s4r_rv.inp	Abaqus/Explicit analysis.

Use of MPCs

Elements tested

CPS4R C3D8R S4R

Problem description

These tests verify the use of multi-point constraints in a sequential import analysis. The models are subjected to monotonically increasing tensile loads. The sequence of tests involves transferring results from Abaqus/Explicit to Abaqus/Standard and then back into Abaqus/Explicit. All tests use CPS4R elements except for the test that uses SLIDER and SS LINEARMPCs. This test uses C3D8R and S4R elements. The material model is not important.

Results and discussion

The results demonstrate that multi-point constraints are transferred correctly between Abaqus/Explicit and Abaqus/Standard.

Input files

LINEARMPC tests:

xs_x_cps4r_mpclin.inp	First Abaqus/Explicit analysis.
xs_s_cps4r_mpclin.inp	Abaqus/Standard analysis.
sx_x_cps4r_mpclin.inp	Second Abaqus/Explicit analysis.

LINKMPC tests:

xs_x_cps4r_mpclink.inp	First Abaqus/Explicit analysis.
xs_s_cps4r_mpclink.inp	Abaqus/Standard analysis.
sx_x_cps4r_mpclink.inp	Second Abaqus/Explicit analysis.

BEAMMPC tests:

xs_x_cps4r_mpcbeam.inp	First Abaqus/Explicit analysis.
xs_s_cps4r_mpcbeam.inp	Abaqus/Standard analysis.
sx_x_cps4r_mpcbeam.inp	Second Abaqus/Explicit analysis.

PINMPC tests:

xs_x_cps4r_mpcpin.inp	First Abaqus/Explicit analysis.
xs_s_cps4r_mpcpin.inp	Abaqus/Standard analysis.
sx_x_cps4r_mpcpin.inp	Second Abaqus/Explicit analysis.

TIEMPC tests:

xs_x_cps4r_mpctie.inp	First Abaqus/Explicit analysis.
xs_s_cps4r_mpctie.inp	Abaqus/Standard analysis.
sx_x_cps4r_mpctie.inp	Second Abaqus/Explicit analysis.

SLIDER and SS LINEARMPC tests:

xs_x_c3d8r_mpcsslin.inp	First Abaqus/Explicit analysis.
xs_s_c3d8r_mpcsslin.inp	Abaqus/Standard analysis.
sx_x_c3d8r_mpcsslin.inp	Second Abaqus/Explicit analysis.

Kinematic and distributing couplings

Elements tested

S4R

Problem description

These tests for kinematic and distributing coupling verify that the displacements and rotations of the reference nodes that do not belong to any imported elements are transferred in the import analysis. The tests use S4R elements and a linear elastic material model and cover Abaqus/Standard to Abaqus/Standard, Abaqus/Standard to Abaqus/Explicit, and Abaqus/Explicit to Abaqus/Explicit import analyses.

Results and discussion

The results demonstrate that the nodal results of the reference nodes of kinematic and distributing couplings are transferred correctly between various Abaqus/Explicit and Abaqus/Standard import analyses.

Input files

collapse_pipe_std.inp	First Abaqus/Standard analysis.
collapse_pipe_std-std.inp	Abaqus/Standard import analysis.
collapse_pipe_std-xpl.inp	Abaqus/Explicit import analysis.
collapse_pipe_xpl.inp	First Abaqus/Explicit analysis.
collapse_pipe_xpl-xpl.inp	Abaqus/Explicit import analysis.

Tie constraints

Elements tested
C3D8R

Problem description

These tests verify the protocol for formation of surface-based tie constraints in an Abaqus/Explicit import analysis. The original configuration is used if a surface-based tie constraint is defined previously for the main and secondary surfaces and the definitions of both surfaces are unchanged in the import analysis. The model consists of blocks separated by an initial gap that are to be tied at locations within various position tolerances. The tests use C3D8R elements and linear elastic and neo-Hookean hyperelastic material models and cover both Abaqus/Standard to Abaqus/Explicit and Abaqus/Explicit to Abaqus/Explicit import analyses.

Results and discussion

In tests where a surface-based tie constraint is defined previously for the surfaces and there is no change in the surface definitions, the elastic blocks that were previously untied remain untied in the import analysis due to use of the original configuration for tie constraint formation. In tests where no tie is defined previously for the surfaces of the elastic blocks or when one of the surfaces is redefined in the import analysis, the blocks that were previously untied become tied due to use of the deformed configuration. The results verify the protocol used for formation of surface-based tie constraints in Abaqus/Explicit import analysis.

Input files

two_blocks_tie_std.inp	First Abaqus/Standard analysis.
two_blocks_tie_std-xpl.inp	Abaqus/Explicit import analysis with UPDATE=NO.
two_blocks_tie_std-xpl_y_y.inp	Abaqus/Explicit import analysis with UPDATE=YES.
two_blocks_tie_std-xpl_new.inp	Abaqus/Explicit import analysis with redefined surface.
two_blocks_notie_std.inp	First Abaqus/Standard analysis in which no surface-based tie constraint is defined for the elastic blocks.
two_blocks_notie_std-xpl.inp	Abaqus/Explicit import analysis with UPDATE=NO from original analysis in which no surface-based tie constraint is defined for the elastic blocks.
two_blocks_tie_xpl.inp	First Abaqus/Explicit analysis.
two_blocks_tie_xpl-xpl.inp	Abaqus/Explicit import analysis with UPDATE=NO.
two_blocks_tie_xpl-xpl_y_y.inp	Abaqus/Explicit import analysis with UPDATE=YES.
two_blocks_tie_xpl-xpl_new.inp	Abaqus/Explicit import analysis with redefined surface.
two_blocks_notie_xpl.inp	First Abaqus/Explicit analysis in which no surface-based tie constraint is defined for the elastic blocks.

two_blocks_notie_xpl-xpl.inp	Abaqus/Explicit import analysis with UPDATE=NO from original analysis in which no surface-based tie constraint is defined for the elastic blocks.

Fluid cavities

Elements tested

SFM3D4R

Problem description

The test for fluid behavior for a surface-based cavity given in *Surface-based fluid cavities* is used to verify transfer of the state at the fluid cavity reference node when the material state is imported. The test uses SFM3D4R elements and covers Abaqus/Explicit to Abaqus/Explicit import analysis. The model used in the import analysis is identical to that of the first analysis in which the response to the applied loading is computed for four steps. In the import analysis, the results from the end of the second step of the first analysis are imported, and the analysis continues for two additional steps using the same loading as given in the first analysis. The results of the third and fourth steps are to be compared in the verification.

Results and discussion

The results at Steps 3 and 4 of the first analysis and the import analysis are the same. This demonstrates that the nodal results of the reference nodes of fluid cavities are transferred correctly between Abaqus/Explicit to Abaqus/Explicit import analyses.

Input files

fluidbehavior.inp	First Abaqus/Explicit analysis.
fluidbehavior_import.inp	Abaqus/Explicit import analysis.

Contact pairs with part instance

Elements tested

S4R

Problem description

The test for transfer of contact conditions given in *Transferring results from one Abaqus/Standard analysis to another Abaqus/Standard analysis* is used to verify transfer of the state of contact pairs for models that use part instances. The test uses S4R elements and covers Abaqus/Standard to Abaqus/Standard import analysis.

The first analysis consists of two blocks of elements initially separated by a small distance. During the first step of the analysis the two blocks are brought together to establish contact. During the second step the blocks are made to slide relative to each other. The material associated with the element is elastic-plastic; the interface between the two blocks has a coefficient of friction of 0.1.

In a second analysis, the results from the end of the first step of the first analysis are transferred, and the two blocks are made to slide relative to each other in a manner identical to that in the second step of the first analysis.

In a third analysis, the second analysis is repeated with the transfer of two instances of the sliding blocks, one in its original position and the other repositioned with a translation of 2 units along the y-axis and a rotation of 90 degrees about the z-axis.

The tests are performed with the material state imported and the reference configuration not updated.

Results and discussion

The results at the end of both import analyses are identical to the results at the end of the first analysis. This demonstrates that the state of the contact pairs is transferred correctly between Abaqus/Standard to Abaqus/Standard import analyses for models that use part instances.

Input files

ss1p_s4r_contact.inp	First Abaqus/Standard analysis.
ss2p_s4r_contact_n_y.inp	Abaqus/Standard import analysis with one instance.
ss2p_s4r_contact_n_y_x2.inp	Abaqus/Standard import analysis with two instances.

Pre-tension section

Elements tested
CPE4R

Problem description

These tests verify that results are imported correctly when a pre-tension section is used in an Abaqus/Standard analysis. Pre-tension loading is applied to the model in Abaqus/Standard; the model is then subjected to tensile loading. The results are imported into Abaqus/Explicit, where additional tension is applied. This result is imported back into Abaqus/Standard, where additional tension is imposed.

Results and discussion

The results demonstrate that the pre-tension section is transferred correctly between Abaqus/Explicit and Abaqus/Standard.

Input files

sx_s_cpe4r_pretens.inp	First Abaqus/Standard analysis.
sx_x_cpe4r_pretens.inp	Abaqus/Explicit analysis.
xs_s_cpe4r_pretens.inp	Second Abaqus/Standard analysis.

Nodal coordinate system

Elements tested
CPE4R M3D4R

Problem description

These tests verify the application of a nodal coordinate system in a sequential import analysis. The nodal coordinate system is redefined in each input file. Two different transformation types are considered: rectangular and cylindrical.

The model using the rectangular transformation is subjected to monotonically increasing tensile loads; the model using the cylindrical transformation is subjected to monotonically increasing torsional loads. The sequence of tests involves transferring results from Abaqus/Standard to Abaqus/Explicit and then back to Abaqus/Standard. The material model is not important.

Results and discussion

The results demonstrate that rectangular and cylindrical transformations are transferred correctly between Abaqus/Explicit and Abaqus/Standard.

Input files

CPE4R element tests:

sx_s_cpe4r_rtrans.inp	First Abaqus/Standard analysis.
sx_x_cpe4r_rtrans.inp	Abaqus/Explicit analysis.
xs_s_cpe4r_rtrans.inp	Second Abaqus/Standard analysis.

M3D4R element tests:

sx_s_m3d4r_ctrans.inp	First Abaqus/Standard analysis.
sx_x_m3d4r_ctrans.inp	Abaqus/Explicit analysis.
xs_s_m3d4r_ctrans.inp	Second Abaqus/Standard analysis.

Steady-state rolling

Elements tested

C3D8R M3D4R S4R

Problem description

These tests verify the transfer of results from Abaqus/Standard to Abaqus/Explicit when steady-state transport is used in Abaqus/Standard. Three input files are used in each verification test. In the first input file an axisymmetric mesh is generated for the cross-section of a disk. The axisymmetric mesh is then used to create a three-dimensional model in the second input file with symmetric model generation. A steady-state rolling analysis is then performed. The steady-state results are imported into Abaqus/Explicit, where the result serves as the initial condition to a transient rolling analysis. Three element types are tested. The following material properties are used (the units are not important):

Young's modulus = 600.

Poisson's ratio = 0.49

Density = 0.036

Results and discussion

The results demonstrate that steady-state transport analyses are transferred correctly between Abaqus/Explicit and Abaqus/Standard.

Input files

C3D8R element tests:

sx_s_c3d8r_ssta.inp	Axisymmetric mesh generation in Abaqus/Standard.
sx_s_c3d8r_sst.inp	Three-dimensional model creation and steady-state rolling analysis in Abaqus/Standard.
sx_x_c3d8r_sst.inp	Transient rolling analysis in Abaqus/Explicit.
sx_x_c3d8r_sst_gcont.inp	Transient rolling analysis using the general contact capability in Abaqus/Explicit.

M3D4R element tests:

sx_s_m3d4r_ssta.inp	Axisymmetric mesh generation in Abaqus/Standard.
sx_s_m3d4r_sst.inp	Three-dimensional model creation and steady-state rolling analysis in Abaqus/Standard.
sx_x_m3d4r_sst.inp	Transient rolling analysis in Abaqus/Explicit.
sx_x_m3d4r_sst_gcont.inp	Transient rolling analysis using the general contact capability in Abaqus/Explicit.

S4R element tests:

sx_s_s4r_ssta.inp	Axisymmetric mesh generation in Abaqus/Standard.
sx_s_s4r_sst.inp	Three-dimensional model creation and steady-state rolling analysis in Abaqus/Standard.
sx_x_s4r_sst.inp	Transient rolling analysis in Abaqus/Explicit.

Transfer of coupled temperature-displacement elements

Elements tested

CAX3T CAX4RT CPE3T CPE4RT CPS3T CPS4RT C3D4T C3D6T C3D8RT C3D8T CAX6MT CPE6MT CPS6MT C3D10MT SC6RT SC8RT S3RT S4RT

Problem description

The tests outlined in this section verify the accuracy of transfer of coupled temperature-displacement elements from Abaqus/Explicit to Abaqus/Standard and vice versa. The tests are performed for each of the elements listed.

The tests for the transfer from Abaqus/Explicit to Abaqus/Standard involve a single element subjected to a combination of thermal loads and prescribed displacements in fully coupled thermal-stress in Abaqus/Explicit. The results from the end of this analysis are then transferred to a fully coupled thermal-stress analysis in Abaqus/Standard in which all the loads on the element are removed and the element is allowed to spring back. Different combinations of the import capability are tested.

The tests for the transfer from Abaqus/Standard to Abaqus/Explicit involve a single element subjected to a combination of thermal loads and prescribed displacements in a fully coupled thermal-stress analysis in Abaqus/Standard. The results from the end of this analysis are then transferred to an Abaqus/Explicit fully coupled thermal-stress analysis in which all the loads on the element are removed so that the element can return to its original undeformed configuration. Different combinations of the import capability are tested.

Results and discussion

The tests demonstrate that the temperature and all state variables, such as the stresses and elastic strains, are transferred accurately from Abaqus/Explicit to Abaqus/Standard and vice versa when the material state is imported. When the reference configuration is updated, the total strains at the beginning of the import analysis are set to zero; when the reference configuration is not updated, the total strains are continuous across the transfer from one analysis code to another.

Input files

Import from Abaqus/Explicit to Abaqus/Standard

CAX3T elements:

xs_x_cax3t.inp	Abaqus/Explicit analysis.
xs_s_cax3t_n_n.inp	Abaqus/Standard import analysis with UPDATE=NO and STATE=NO.

xs_s_cax3t_n_y.inp	Abaqus/Standard import analysis with UPDATE=NO and STATE=YES.
xs_s_cax3t_y_n.inp	Abaqus/Standard import analysis with UPDATE=YES and STATE=NO.
xs_s_cax3t_y_y.inp	Abaqus/Standard import analysis with UPDATE=YES and STATE=YES.

CAX4RT elements:

xs_x_cax4rt.inp	Abaqus/Explicit analysis.
xs_s_cax4rt_n_n.inp	Abaqus/Standard import analysis with UPDATE=NO and STATE=NO.
xs_s_cax4rt_n_y.inp	Abaqus/Standard import analysis with UPDATE=NO and STATE=YES.
xs_s_cax4rt_y_n.inp	Abaqus/Standard import analysis with UPDATE=YES and STATE=NO.
xs_s_cax4rt_y_y.inp	Abaqus/Standard import analysis with UPDATE=YES and STATE=YES.

CPE3T elements:

xs_x_cpe3t.inp	Abaqus/Explicit analysis.
xs_s_cpe3t_n_n.inp	Abaqus/Standard import analysis with UPDATE=NO and STATE=NO.
xs_s_cpe3t_n_y.inp	Abaqus/Standard import analysis with UPDATE=NO and STATE=YES.
xs_s_cpe3t_y_n.inp	Abaqus/Standard import analysis with UPDATE=YES and STATE=NO.
xs_s_cpe3t_y_y.inp	Abaqus/Standard import analysis with UPDATE=YES and STATE=YES.

CPE4RT elements:

xs_x_cpe4rt.inp	Abaqus/Explicit analysis.
xs_s_cpe4rt_n_n.inp	Abaqus/Standard import analysis with UPDATE=NO and STATE=NO.
xs_s_cpe4rt_n_y.inp	Abaqus/Standard import analysis with UPDATE=NO and STATE=YES.
xs_s_cpe4rt_y_n.inp	Abaqus/Standard import analysis with UPDATE=YES and STATE=NO.
xs_s_cpe4rt_y_y.inp	Abaqus/Standard import analysis with UPDATE=YES and STATE=YES.

CPS3T elements:

xs_x_cps3t.inp	Abaqus/Explicit analysis.
xs_s_cps3t_n_n.inp	Abaqus/Standard import analysis with UPDATE=NO and STATE=NO.
xs_s_cps3t_n_y.inp	Abaqus/Standard import analysis with UPDATE=NO and STATE=YES.
xs_s_cps3t_y_n.inp	Abaqus/Standard import analysis with UPDATE=YES and STATE=NO.
xs_s_cps3t_y_y.inp	Abaqus/Standard import analysis with UPDATE=YES and STATE=YES.

CPS4RT elements:

xs_x_cps4rt.inp	Abaqus/Explicit analysis.
xs_s_cps4rt_n_n.inp	Abaqus/Standard import analysis with UPDATE=NO and STATE=NO.
xs_s_cps4rt_n_y.inp	Abaqus/Standard import analysis with UPDATE=NO and STATE=YES.
xs_s_cps4rt_y_n.inp	Abaqus/Standard import analysis with UPDATE=YES and STATE=NO.

xs_s_cps4rt_y_y.inp	Abaqus/Standard import analysis with UPDATE=YES and STATE=YES.

C3D4T elements:

xs_x_c3d4t.inp	Abaqus/Explicit analysis.
xs_s_c3d4t_n_n.inp	Abaqus/Standard import analysis with UPDATE=NO and STATE=NO.
xs_s_c3d4t_n_y.inp	Abaqus/Standard import analysis with UPDATE=NO and STATE=YES.
xs_s_c3d4t_y_n.inp	Abaqus/Standard import analysis with UPDATE=YES and STATE=NO.
xs_s_c3d4t_y_y.inp	Abaqus/Standard import analysis with UPDATE=YES and STATE=YES.

C3D6T elements:

xs_x_c3d6t.inp	Abaqus/Explicit analysis.
xs_s_c3d6t_n_n.inp	Abaqus/Standard import analysis with UPDATE=NO and STATE=NO.
xs_s_c3d6t_n_y.inp	Abaqus/Standard import analysis with UPDATE=NO and STATE=YES.
xs_s_c3d6t_y_n.inp	Abaqus/Standard import analysis with UPDATE=YES and STATE=NO.
xs_s_c3d6t_y_y.inp	Abaqus/Standard import analysis with UPDATE=YES and STATE=YES.

C3D8RT elements:

xs_x_c3d8rt.inp	Abaqus/Explicit analysis.
xs_s_c3d8rt_n_n.inp	Abaqus/Standard import analysis with UPDATE=NO and STATE=NO.
xs_s_c3d8rt_n_y.inp	Abaqus/Standard import analysis with UPDATE=NO and STATE=YES.
xs_s_c3d8rt_y_n.inp	Abaqus/Standard import analysis with UPDATE=YES and STATE=NO.
xs_s_c3d8rt_y_y.inp	Abaqus/Standard import analysis with UPDATE=YES and STATE=YES.

C3D8T elements:

xs_x_c3d8t.inp	Abaqus/Explicit analysis.
xs_s_c3d8t_n_n.inp	Abaqus/Standard import analysis with UPDATE=NO and STATE=NO.
xs_s_c3d8t_n_y.inp	Abaqus/Standard import analysis with UPDATE=NO and STATE=YES.
xs_s_c3d8t_y_n.inp	Abaqus/Standard import analysis with UPDATE=YES and STATE=NO.
xs_s_c3d8t_y_y.inp	Abaqus/Standard import analysis with UPDATE=YES and STATE=YES.

CAX6MT elements:

xs_x_cax6mt.inp	Abaqus/Explicit analysis.
xs_s_cax6mt_n_n.inp	Abaqus/Standard import analysis with UPDATE=NO and STATE=NO.
xs_s_cax6mt_n_y.inp	Abaqus/Standard import analysis with UPDATE=NO and STATE=YES.
xs_s_cax6mt_y_n.inp	Abaqus/Standard import analysis with UPDATE=YES and STATE=NO.
xs_s_cax6mt_y_y.inp	Abaqus/Standard import analysis with UPDATE=YES and STATE=YES.

CPE6MT elements:

xs_x_cpe6mt.inp	Abaqus/Explicit analysis.
xs_s_cpe6mt_n_n.inp	Abaqus/Standard import analysis with UPDATE=NO and STATE=NO.
xs_s_cpe6mt_n_y.inp	Abaqus/Standard import analysis with UPDATE=NO and STATE=YES.
xs_s_cpe6mt_y_n.inp	Abaqus/Standard import analysis with UPDATE=YES and STATE=NO.
xs_s_cpe6mt_y_y.inp	Abaqus/Standard import analysis with UPDATE=YES and STATE=YES.

CPS6MT elements:

xs_x_cps6mt.inp	Abaqus/Explicit analysis.
xs_s_cps6mt_n_n.inp	Abaqus/Standard import analysis with UPDATE=NO and STATE=NO.
xs_s_cps6mt_n_y.inp	Abaqus/Standard import analysis with UPDATE=NO and STATE=YES.
xs_s_cps6mt_y_n.inp	Abaqus/Standard import analysis with UPDATE=YES and STATE=NO.
xs_s_cps6mt_y_y.inp	Abaqus/Standard import analysis with UPDATE=YES and STATE=YES.

C3D10MT elements:

xs_x_c3d10mt.inp	Abaqus/Explicit analysis.
xs_s_c3d10mt_n_n.inp	Abaqus/Standard import analysis with UPDATE=NO and STATE=NO.
xs_s_c3d10mt_n_y.inp	Abaqus/Standard import analysis with UPDATE=NO and STATE=YES.
xs_s_c3d10mt_y_n.inp	Abaqus/Standard import analysis with UPDATE=YES and STATE=NO.
xs_s_c3d10mt_y_y.inp	Abaqus/Standard import analysis with UPDATE=YES and STATE=YES.

SC6RT elements:

xs_x_sc6rt.inp	Abaqus/Explicit analysis.
xs_s_sc6rt_n_n.inp	Abaqus/Standard import analysis with UPDATE=NO and STATE=NO.
xs_s_sc6rt_n_y.inp	Abaqus/Standard import analysis with UPDATE=NO and STATE=YES.
xs_s_sc6rt_y_n.inp	Abaqus/Standard import analysis with UPDATE=YES and STATE=NO.
xs_s_sc6rt_y_y.inp	Abaqus/Standard import analysis with UPDATE=YES and STATE=YES.

SC8RT elements:

xs_x_sc8rt.inp	Abaqus/Explicit analysis.
xs_s_sc8rt_n_n.inp	Abaqus/Standard import analysis with UPDATE=NO and STATE=NO.
xs_s_sc8rt_n_y.inp	Abaqus/Standard import analysis with UPDATE=NO and STATE=YES.
xs_s_sc8rt_y_n.inp	Abaqus/Standard import analysis with UPDATE=YES and STATE=NO.
xs_s_sc8rt_y_y.inp	Abaqus/Standard import analysis with UPDATE=YES and STATE=YES.

S3RT elements:

xs_x_s3rt.inp	Abaqus/Explicit analysis.

xs_s_s3rt_n_n.inp	Abaqus/Standard import analysis with UPDATE=NO and STATE=NO.
xs_s_s3rt_n_y.inp	Abaqus/Standard import analysis with UPDATE=NO and STATE=YES.
xs_s_s3rt_y_n.inp	Abaqus/Standard import analysis with UPDATE=YES and STATE=NO.
xs_s_s3rt_y_y.inp	Abaqus/Standard import analysis with UPDATE=YES and STATE=YES.

S4RT elements:

xs_x_s4rt.inp	Abaqus/Explicit analysis.
xs_s_s4rt_n_n.inp	Abaqus/Standard import analysis with UPDATE=NO and STATE=NO.
xs_s_s4rt_n_y.inp	Abaqus/Standard import analysis with UPDATE=NO and STATE=YES.
xs_s_s4rt_y_n.inp	Abaqus/Standard import analysis with UPDATE=YES and STATE=NO.
xs_s_s4rt_y_y.inp	Abaqus/Standard import analysis with UPDATE=YES and STATE=YES.

Import from Abaqus/Standard to Abaqus/Explicit

CAX3T elements:

sx_s_cax3t.inp	Abaqus/Standard analysis.
sx_x_cax3t_n_n.inp	Abaqus/Explicit import analysis with UPDATE=NO and STATE=NO.
sx_x_cax3t_n_y.inp	Abaqus/Explicit import analysis with UPDATE=NO and STATE=YES.
sx_x_cax3t_y_n.inp	Abaqus/Explicit import analysis with UPDATE=YES and STATE=NO.
sx_x_cax3t_y_y.inp	Abaqus/Explicit import analysis with UPDATE=YES and STATE=YES.

CAX4RT elements:

sx_s_cax4rt.inp	Abaqus/Standard analysis.
sx_x_cax4rt_n_n.inp	Abaqus/Explicit import analysis with UPDATE=NO and STATE=NO.
sx_x_cax4rt_n_y.inp	Abaqus/Explicit import analysis with UPDATE=NO and STATE=YES.
sx_x_cax4rt_y_n.inp	Abaqus/Explicit import analysis with UPDATE=YES and STATE=NO.
sx_x_cax4rt_y_y.inp	Abaqus/Explicit import analysis with UPDATE=YES and STATE=YES.

CPE3T elements:

sx_s_cpe3t.inp	Abaqus/Standard analysis.
sx_x_cpe3t_n_n.inp	Abaqus/Explicit import analysis with UPDATE=NO and STATE=NO.
sx_x_cpe3t_n_y.inp	Abaqus/Explicit import analysis with UPDATE=NO and STATE=YES.
sx_x_cpe3t_y_n.inp	Abaqus/Explicit import analysis with UPDATE=YES and STATE=NO.
sx_x_cpe3t_y_y.inp	Abaqus/Explicit import analysis with UPDATE=YES and STATE=YES.

CPE4RT elements:

sx_s_cpe4rt.inp	Abaqus/Standard analysis.
sx_x_cpe4rt_n_n.inp	Abaqus/Explicit import analysis with UPDATE=NO and STATE=NO.
sx_x_cpe4rt_n_y.inp	Abaqus/Explicit import analysis with UPDATE=NO and STATE=YES.
sx_x_cpe4rt_y_n.inp	Abaqus/Explicit import analysis with UPDATE=YES and STATE=NO.
sx_x_cpe4rt_y_y.inp	Abaqus/Explicit import analysis with UPDATE=YES and STATE=YES.

CPS3T elements:

sx_s_cps3t.inp	Abaqus/Standard analysis.
sx_x_cps3t_n_n.inp	Abaqus/Explicit import analysis with UPDATE=NO and STATE=NO.
sx_x_cps3t_n_y.inp	Abaqus/Explicit import analysis with UPDATE=NO and STATE=YES.
sx_x_cps3t_y_n.inp	Abaqus/Explicit import analysis with UPDATE=YES and STATE=NO.
sx_x_cps3t_y_y.inp	Abaqus/Explicit import analysis with UPDATE=YES and STATE=YES.

CPS4RT elements:

sx_s_cps4rt.inp	Abaqus/Standard analysis.
sx_x_cps4rt_n_n.inp	Abaqus/Explicit import analysis with UPDATE=NO and STATE=NO.
sx_x_cps4rt_n_y.inp	Abaqus/Explicit import analysis with UPDATE=NO and STATE=YES.
sx_x_cps4rt_y_n.inp	Abaqus/Explicit import analysis with UPDATE=YES and STATE=NO.
sx_x_cps4rt_y_y.inp	Abaqus/Explicit import analysis with UPDATE=YES and STATE=YES.

C3D4T elements:

sx_s_c3d4t.inp	Abaqus/Standard analysis.
sx_x_c3d4t_n_n.inp	Abaqus/Explicit import analysis with UPDATE=NO and STATE=NO.
sx_x_c3d4t_n_y.inp	Abaqus/Explicit import analysis with UPDATE=NO and STATE=YES.
sx_x_c3d4t_y_n.inp	Abaqus/Explicit import analysis with UPDATE=YES and STATE=NO.
sx_x_c3d4t_y_y.inp	Abaqus/Explicit import analysis with UPDATE=YES and STATE=YES.

C3D6T elements:

sx_s_c3d6t.inp	Abaqus/Standard analysis.
sx_x_c3d6t_n_n.inp	Abaqus/Explicit import analysis with UPDATE=NO and STATE=NO.
sx_x_c3d6t_n_y.inp	Abaqus/Explicit import analysis with UPDATE=NO and STATE=YES.
sx_x_c3d6t_y_n.inp	Abaqus/Explicit import analysis with UPDATE=YES and STATE=NO.
sx_x_c3d6t_y_y.inp	Abaqus/Explicit import analysis with UPDATE=YES and STATE=YES.

C3D8RT elements:

sx_s_c3d8rt.inp	Abaqus/Standard analysis.
sx_x_c3d8rt_n_n.inp	Abaqus/Explicit import analysis with UPDATE=NO and STATE=NO.
sx_x_c3d8rt_n_y.inp	Abaqus/Explicit import analysis with UPDATE=NO and STATE=YES.
sx_x_c3d8rt_y_n.inp	Abaqus/Explicit import analysis with UPDATE=YES and STATE=NO.
sx_x_c3d8rt_y_y.inp	Abaqus/Explicit import analysis with UPDATE=YES and STATE=YES.

CAX6MT elements:

sx_s_cax6mt.inp	Abaqus/Standard analysis.
sx_x_cax6mt_n_n.inp	Abaqus/Explicit import analysis with UPDATE=NO and STATE=NO.
sx_x_cax6mt_n_y.inp	Abaqus/Explicit import analysis with UPDATE=NO and STATE=YES.
sx_x_cax6mt_y_n.inp	Abaqus/Explicit import analysis with UPDATE=YES and STATE=NO.

sx_x_cax6mt_y_y.inp	Abaqus/Explicit import analysis with UPDATE=YES and STATE=YES.

CPE6MT elements:

sx_s_cpe6mt.inp	Abaqus/Standard analysis.
sx_x_cpe6mt_n_n.inp	Abaqus/Explicit import analysis with UPDATE=NO and STATE=NO.
sx_x_cpe6mt_n_y.inp	Abaqus/Explicit import analysis with UPDATE=NO and STATE=YES.
sx_x_cpe6mt_y_n.inp	Abaqus/Explicit import analysis with UPDATE=YES and STATE=NO.
sx_x_cpe6mt_y_y.inp	Abaqus/Explicit import analysis with UPDATE=YES and STATE=YES.

CPS6MT elements:

sx_s_cps6mt.inp	Abaqus/Standard analysis.
sx_x_cps6mt_n_n.inp	Abaqus/Explicit import analysis with UPDATE=NO and STATE=NO.
sx_x_cps6mt_n_y.inp	Abaqus/Explicit import analysis with UPDATE=NO and STATE=YES.
sx_x_cps6mt_y_n.inp	Abaqus/Explicit import analysis with UPDATE=YES and STATE=NO.
sx_x_cps6mt_y_y.inp	Abaqus/Explicit import analysis with UPDATE=YES and STATE=YES.

C3D10MT elements:

sx_s_c3d10mt.inp	Abaqus/Standard analysis.
sx_x_c3d10mt_n_n.inp	Abaqus/Explicit import analysis with UPDATE=NO and STATE=NO.
sx_x_c3d10mt_n_y.inp	Abaqus/Explicit import analysis with UPDATE=NO and STATE=YES.
sx_x_c3d10mt_y_n.inp	Abaqus/Explicit import analysis with UPDATE=YES and STATE=NO.
sx_x_c3d10mt_y_y.inp	Abaqus/Explicit import analysis with UPDATE=YES and STATE=YES.

Transfer of rebar layers and embedded elements

Elements tested

C3D8R M3D3 M3D4R M3D4 S3R S4R SAX1 SFM3D3 SFM3D4R

Problem description

The tests outlined in this section verify the accuracy of the transfer of rebar layers and embedded elements from Abaqus/Explicit to Abaqus/Standard and vice versa. The tests are performed for each of the elements listed.

The tests for the transfer from Abaqus/Explicit to Abaqus/Standard involve elements with rebar layers or embedded elements subjected to loading over two explicit dynamic steps. The results from the end of the first step are then transferred to an Abaqus/Standard static import analysis. In addition to the imported elements, new elements with rebar layers or embedded elements are defined in the import analysis. These new elements are identical to the initial element definitions of the imported elements in the original Abaqus/Explicit analysis. The import

analysis is performed for the different combinations in which the material state is imported and the reference configuration is considered as both updated and not updated.

The tests for the transfer from Abaqus/Standard to Abaqus/Explicit involve elements with rebar layers or embedded elements subjected to loading over two static steps. The results from the end of the first step are then transferred to an Abaqus/Explicit dynamic import analysis. In addition to the imported elements, new elements with rebar layers or embedded elements are defined in the import analysis. These new elements are identical to the initial element definitions of the imported elements in the original Abaqus/Standard analysis. The import analysis is performed for the different combinations in which the material state is imported and the reference configuration is considered as both updated and not updated.

Results and discussion

The results for the two sets of elements in the import analysis (that is, the newly defined elements and the imported elements) are identical at the end of the analysis when the material state is imported and the reference configuration is not updated. In addition, these results are in good agreement with the results at the end of the second step of the original analysis. These tests demonstrate that appropriate quantities in the rebar layer and embedded elements (such as the stresses, rebar orientations, strains, etc.) are transferred accurately from Abaqus/Explicit to Abaqus/Standard and vice versa. The only differences in the results at the end of the import analysis when the reference configuration is updated compared to the results when the reference configuration is not updated are in the kinematic quantities such as the total strains, rebar rotations, etc. When the reference configuration is updated in the import analysis, the total strains and the rebar rotations at the beginning of the import analysis are set to zero; when the reference configuration is not updated, the total strains and the rebar rotations are continuous across the transfer from one analysis code to another.

Input files

Import from Abaqus/Explicit to Abaqus/Standard

xs_x_rebar_memb.inp	Abaqus/Explicit analysis.
xs_s_rebar_memb_n_y.inp	Abaqus/Standard import analysis with UPDATE=NO and STATE=YES.
xs_s_rebar_memb_y_y.inp	Abaqus/Standard import analysis with UPDATE=YES and STATE=YES.
xs_x_rebar_memb_embed.inp	Abaqus/Explicit analysis.
xs_s_rebar_memb_embed_n_y.inp	Abaqus/Standard import analysis with UPDATE=NO and STATE=YES.
xs_s_rebar_memb_embed_y_y.inp	Abaqus/Standard import analysis with UPDATE=YES and STATE=YES.
xs_s_rebar_memb_embed_y_n.inp	Abaqus/Standard import analysis with UPDATE=YES and STATE=NO.
xs_x_rebar_m3d4_embed.inp	Abaqus/Explicit analysis.
xs_s_rebar_m3d4_embed_n_y.inp	Abaqus/Standard import analysis with UPDATE=NO and STATE=YES.
xs_s_rebar_m3d4_embed_y_y.inp	Abaqus/Standard import analysis with UPDATE=YES and STATE=YES.
xs_s_rebar_m3d4_embed_y_n.inp	Abaqus/Standard import analysis with UPDATE=YES and STATE=NO.
xs_x_rebar_shell.inp	Abaqus/Explicit analysis.

xs_s_rebar_shell_n_y.inp	Abaqus/Standard import analysis with UPDATE=NO and STATE=YES.
xs_s_rebar_shell_y_y.inp	Abaqus/Standard import analysis with UPDATE=YES and STATE=YES.
xs_x_rebar_shellax.inp	Abaqus/Explicit analysis.
xs_s_rebar_shellax_n_y.inp	Abaqus/Standard import analysis with UPDATE=NO and STATE=YES.
xs_s_rebar_shellax_y_y.inp	Abaqus/Standard import analysis with UPDATE=YES and STATE=YES.
xs_x_rebar_surf.inp	Abaqus/Explicit analysis.
xs_s_rebar_surf_n_y.inp	Abaqus/Standard import analysis with UPDATE=NO and STATE=YES.
xs_s_rebar_surf_y_y.inp	Abaqus/Standard import analysis with UPDATE=YES and STATE=YES.

Import from Abaqus/Standard to Abaqus/Explicit

sx_s_rebar_memb.inp	Abaqus/Standard analysis.
sx_x_rebar_memb_n_y.inp	Abaqus/Explicit import analysis with UPDATE=NO and STATE=YES.
sx_x_rebar_memb_y_y.inp	Abaqus/Explicit import analysis with UPDATE=YES and STATE=YES.
sx_s_rebar_memb_embed.inp	Abaqus/Standard analysis.
sx_x_rebar_memb_embed_n_y.inp	Abaqus/Explicit import analysis with UPDATE=NO and STATE=YES.
sx_x_rebar_memb_embed_y_y.inp	Abaqus/Explicit import analysis with UPDATE=YES and STATE=YES.
sx_s_rebar_shell.inp	Abaqus/Standard analysis.
sx_x_rebar_shell_n_y.inp	Abaqus/Explicit import analysis with UPDATE=NO and STATE=YES.
sx_x_rebar_shell_y_y.inp	Abaqus/Explicit import analysis with UPDATE=YES and STATE=YES.
sx_s_rebar_shellax.inp	Abaqus/Standard analysis.
sx_x_rebar_shellax_n_y.inp	Abaqus/Explicit import analysis with UPDATE=NO and STATE=YES.
sx_x_rebar_shellax_y_y.inp	Abaqus/Explicit import analysis with UPDATE=YES and STATE=YES.
sx_s_rebar_surf.inp	Abaqus/Standard analysis.
sx_x_rebar_surf_n_y.inp	Abaqus/Explicit import analysis with UPDATE=NO and STATE=YES.
sx_x_rebar_surf_y_y.inp	Abaqus/Explicit import analysis with UPDATE=YES and STATE=YES.

Series of transfers between Abaqus/Explicit and Abaqus/Standard

Elements tested

C3D4 C3D6 C3D8R CPE3 CPE4R CPS3 CPS4R S4R

Problem description

The tests outlined in this section verify the transfer of results between Abaqus analysis products by performing a series of transfers between Abaqus/Explicit and Abaqus/Standard and also from one Abaqus/Standard analysis to another Abaqus/Standard analysis using the import capability. The finite element model for each test is a cantilever beam composed of the element types listed and subjected to a series of loading and springback steps in both Abaqus/Standard and Abaqus/Explicit. The transfer of results from one analysis to another is verified. All the tests use the import capability with the material state imported and the reference configuration not updated.

The material used in each test is isotropic linear elasticity, together with Mises plasticity. The material properties used are (the units are not important):

Young's modulus = 200E9.

Poisson's ratio = 0.3

Yield strength= 380E6

Results and discussion

These tests confirm that the results from the end of each analysis are accurately transferred to the subsequent import analysis.

Input files

C3D4 element tests:

ssx1_c3d4_cb.inp	Abaqus/Standard analysis; cantilever beam is subjected to bending.
ssx2_c3d4_cb.inp	Abaqus/Standard import analysis; springback.
ssx3_c3d4_cb.inp	Abaqus/Explicit import analysis; cantilever beam is subjected to bending.
ssx4_c3d4_cb.inp	Abaqus/Standard import analysis; springback.
ssx5_c3d4_cb.inp	Abaqus/Standard import analysis; cantilever beam is subjected to extension.
ssx6_c3d4_cb.inp	Abaqus/Standard import analysis; springback.
ssx7_c3d4_cb.inp	Abaqus/Explicit import analysis; cantilever beam is subjected to twisting.
ssx8_c3d4_cb.inp	Abaqus/Standard import analysis; springback.

C3D6 element tests:

ssx1_c3d6_cb.inp	Abaqus/Standard analysis; cantilever beam is subjected to bending.
ssx2_c3d6_cb.inp	Abaqus/Standard import analysis; springback.
ssx3_c3d6_cb.inp	Abaqus/Explicit import analysis; cantilever beam is subjected to bending.
ssx4_c3d6_cb.inp	Abaqus/Standard import analysis; springback.
ssx5_c3d6_cb.inp	Abaqus/Standard import analysis; cantilever beam is subjected to extension.

ssx6_c3d6_cb.inp	Abaqus/Standard import analysis; springback.
ssx7_c3d6_cb.inp	Abaqus/Explicit import analysis; cantilever beam is subjected to twisting.
ssx8_c3d6_cb.inp	Abaqus/Standard import analysis; springback.

C3D8R element tests:

ssx1_c3d8r_cb.inp	Abaqus/Standard analysis; cantilever beam is subjected to bending.
ssx2_c3d8r_cb.inp	Abaqus/Standard import analysis; springback.
ssx3_c3d8r_cb.inp	Abaqus/Explicit import analysis; cantilever beam is subjected to bending.
ssx4_c3d8r_cb.inp	Abaqus/Standard import analysis; springback.
ssx5_c3d8r_cb.inp	Abaqus/Standard import analysis; cantilever beam is subjected to extension.
ssx6_c3d8r_cb.inp	Abaqus/Standard import analysis; springback.
ssx7_c3d8r_cb.inp	Abaqus/Explicit import analysis; cantilever beam is subjected to twisting.
ssx8_c3d8r_cb.inp	Abaqus/Standard import analysis; springback.

CPE3 element tests:

ssx1_cpe3_cb.inp	Abaqus/Standard analysis; cantilever beam is subjected to bending.
ssx2_cpe3_cb.inp	Abaqus/Standard import analysis; springback.
ssx3_cpe3_cb.inp	Abaqus/Explicit import analysis; cantilever beam is subjected to bending.
ssx4_cpe3_cb.inp	Abaqus/Standard import analysis; springback.
ssx5_cpe3_cb.inp	Abaqus/Standard import analysis; cantilever beam is subjected to extension.
ssx6_cpe3_cb.inp	Abaqus/Standard import analysis; springback.

CPE4R element tests:

ssx1_cpe4r_cb.inp	Abaqus/Standard analysis; cantilever beam is subjected to bending.
ssx2_cpe4r_cb.inp	Abaqus/Standard import analysis; springback.
ssx3_cpe4r_cb.inp	Abaqus/Explicit import analysis; cantilever beam is subjected to bending.
ssx4_cpe4r_cb.inp	Abaqus/Standard import analysis; springback.
ssx5_cpe4r_cb.inp	Abaqus/Standard import analysis; cantilever beam is subjected to extension.
ssx6_cpe4r_cb.inp	Abaqus/Standard import analysis; springback.

CPS3 element tests:

ssx1_cps3_cb.inp	Abaqus/Standard analysis; cantilever beam is subjected to bending.
ssx2_cps3_cb.inp	Abaqus/Standard import analysis; springback.
ssx3_cps3_cb.inp	Abaqus/Explicit import analysis; cantilever beam is subjected to bending.
ssx4_cps3_cb.inp	Abaqus/Standard import analysis; springback.
ssx5_cps3_cb.inp	Abaqus/Standard import analysis; cantilever beam is subjected to extension.
ssx6_cps3_cb.inp	Abaqus/Standard import analysis; springback.

CPS4R element tests:

ssx1_cps4r_cb.inp	Abaqus/Standard analysis; cantilever beam is subjected to bending.
ssx2_cps4r_cb.inp	Abaqus/Standard import analysis; springback.
ssx3_cps4r_cb.inp	Abaqus/Explicit import analysis; cantilever beam is subjected to bending.
ssx4_cps4r_cb.inp	Abaqus/Standard import analysis; springback.
ssx5_cps4r_cb.inp	Abaqus/Standard import analysis; cantilever beam is subjected to extension.
ssx6_cps4r_cb.inp	Abaqus/Standard import analysis; springback.

S4R element tests:

ssx1_s4r_cb.inp	Abaqus/Standard analysis; cantilever beam is subjected to bending.
ssx2_s4r_cb.inp	Abaqus/Standard import analysis; springback.
ssx3_s4r_cb.inp	Abaqus/Explicit import analysis; cantilever beam is subjected to bending.
ssx4_s4r_cb.inp	Abaqus/Standard import analysis; springback.
ssx5_s4r_cb.inp	Abaqus/Standard import analysis; cantilever beam is subjected to extension.
ssx6_s4r_cb.inp	Abaqus/Standard import analysis; springback.
ssx7_s4r_cb.inp	Abaqus/Explicit import analysis; cantilever beam is subjected to twisting.
ssx8_s4r_cb.inp	Abaqus/Standard import analysis; springback.

Transferring results with parallel rheological framework

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

CAX4H CAX4R CPE4H CPE4R C3D8R

Problem description

The verification tests in this section consist of one-element models that are subjected to uniaxial loads in a sequential import analysis. The material model is defined using the parallel rheological framework and consists of an elastic-plastic equilibrium network with Mullins effect and three nonlinear viscoelastic networks. The viscous response in the viscoelastic networks is modeled using the power law, strain hardening, and hyperbolic-sine creep models. Three different strain energy potentials, neo-Hookean, Ogden, and Arruda-Boyce, are considered. The results are transferred from the middle of the analysis and are allowed to continue in the import analysis.

Results and discussion

The results demonstrate that the material models defined using the parallel rheological framework transferred successfully between Abaqus analyses.

Input files

Transferring results from Abaqus/Standard to Abaqus/Standard:

ss_prf_base.inp	Abaqus/Standard analysis, viscoelastic-plastic material with three networks.
ss_prf_import_ny.inp	Abaqus/Standard import analysis, UPDATE=NO, STATE=YES.
ss_prf_import_yy.inp	Abaqus/Standard import analysis, UPDATE=YES, STATE=YES.
ss_prf_import_yn.inp	Abaqus/Standard import analysis, UPDATE=YES, STATE=NO.

Transferring results from Abaqus/Standard to Abaqus/Explicit:

sx_prf_base.inp	Abaqus/Standard analysis, viscoelastic-plastic material with three networks.
sx_prf_import_ny.inp	Abaqus/Explicit import analysis, UPDATE=NO, STATE=YES.
sx_prf_import_yy.inp	Abaqus/Explicit import analysis, UPDATE=YES, STATE=YES.
sx_prf_import_yn.inp	Abaqus/Explicit import analysis, UPDATE=YES, STATE=NO.

Transferring results from Abaqus/Explicit to Abaqus/Explicit:

xx_prf_base.inp	Abaqus/Explicit analysis, viscoelastic-plastic material with three networks.
xx_prf_import_ny.inp	Abaqus/Explicit import analysis, UPDATE=NO, STATE=YES.
xx_prf_import_yy.inp	Abaqus/Explicit import analysis, UPDATE=YES, STATE=YES.
xx_prf_import_yn.inp	Abaqus/Explicit import analysis, UPDATE=YES, STATE=NO.

Transferring results from Abaqus/Explicit to Abaqus/Standard:

xs_prf_base.inp	Abaqus/Explicit analysis, viscoelastic-plastic material with three networks.
xs_prf_import_ny.inp	Abaqus/Standard import analysis, UPDATE=NO, STATE=YES.
xs_prf_import_yy.inp	Abaqus/Standard import analysis, UPDATE=YES, STATE=YES.
xs_prf_import_yn.inp	Abaqus/Standard import analysis, UPDATE=YES, STATE=NO.

Transferring results with nodal temperature and field variables

Products: Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Transferring results with nodal temperature and field variables from an Abaqus/Standard analysis to an Abaqus/Explicit analysis

Elements tested

B31 C3D4 C3D6 C3D8 C3D8I C3D8R C3D10M
CPE3 CPE4R CPE6M CPS3 CPS4R CPS6M
M3D3 M3D4R S3R S4R SC8R

Problem description

The verification problems presented in this section test the transfer of results with nodal temperature and field variables from an Abaqus/Standard analysis to an Abaqus/Explicit analysis.

The model in each test consists of disjoint blocks of various types of elements. The blocks are fixed at one end and subject to concentrated loads at the other. The elements reference an elastic material for which the properties are given as tabulated data of the temperature and field variables 1 and 2. In Step 1 the static response is computed in an Abaqus/Standard analysis with given values of concentrated loads, nodal temperature, and nodal field variables. In Step 2 the model and the results at the end of Step 1 are imported to an Abaqus/Explicit analysis and the solution continues with the same load, nodal temperature, and field variable values used in Step 1. The nodal and elemental results of Step 2 in the import analysis are compared to those of Step 1 in the previous analysis.

Results and discussion

The nodal and elemental results in Step 2 of the import analysis are the same as those in Step 1 of the previous analysis. These tests verify that an analysis model with nodal temperature and field variables can be imported from an Abaqus/Standard analysis into an Abaqus/Explicit analysis.

Input files

Planar element tests:

solid_2d_elast_tfv_std.inp	First Abaqus/Standard analysis.
solid_2d_elast_tfv_std-xpl.inp	Second Abaqus/Explicit import analysis.
solid_2d_elast_tfv_model_std-xpl.inp	Second Abaqus/Explicit import analysis with full model input.

Three-dimensional solid element tests:

solid_3d_elast_tfv_std.inp	First Abaqus/Standard analysis.
solid_3d_elast_tfv_std-xpl.inp	Second Abaqus/Explicit import analysis.
solid_3d_elast_tfv_default_std-xpl.inp	Second Abaqus/Explicit import analysis with default all section input.
solid_3d_elast_tfv_model_std-xpl.inp	Second Abaqus/Explicit import analysis with full model input.

solid_3d_elast_tfv_model_default_std-xpl.inp	Second Abaqus/Explicit import analysis with full model and default all section input.

Three-dimensional solid and structural element tests:

solid_str_elast_tfv_std.inp	First Abaqus/Standard analysis.
solid_str_elast_tfv_std-xpl.inp	Second Abaqus/Explicit import analysis.
solid_str_elast_tfv_default_std-xpl.inp	Second Abaqus/Explicit import analysis with default all section input.
solid_str_elast_tfv_model_std-xpl.inp	Second Abaqus/Explicit import analysis with full model input.
solid_str_elast_tfv_model_default_std-xpl.inp	Second Abaqus/Explicit import analysis with full model and default all section input.

Transferring results with nodal temperature and field variables from an Abaqus/Standard analysis to an Abaqus/Standard analysis

Elements tested

C3D4 C3D6 C3D8 C3D8H C3D8I C3D8R C3D10M
CPE3 CPE4R CPE6M CPS3 CPS4R CPS6M
M3D3 M3D4R S3R S4R SC8R

Problem description

The verification problems presented in this section test the transfer of results with nodal temperature and field variables from an Abaqus/Standard analysis to an Abaqus/Standard analysis.

The model in each test consists of disjoint blocks of various types of elements. The blocks are fixed at one end and subject to concentrated loads at the other. The elements reference an elastic material for which the properties are given as tabulated data of the temperature and field variables 1 and 2. In Step 1 the static response is computed in an Abaqus/Standard analysis with given values of concentrated loads, nodal temperature, and nodal field variables. In Step 2 the model and the results at the end of Step 1 are imported to an Abaqus/Standard static analysis and the solution continues with the same load, nodal temperature, and field variable values used in Step 1. The nodal and elemental results of Step 2 in the import analysis are compared to those of Step 1 in the previous analysis.

Results and discussion

The nodal and elemental results in Step 2 of the import analysis are the same as those in Step 1 of the previous analysis. These tests verify that an analysis model with nodal temperature and field variables can be imported from an Abaqus/Standard analysis into an Abaqus/Standard analysis.

Input files

Planar element tests:

solid_2d_elast_tfv_std.inp	First Abaqus/Standard analysis.
solid_2d_elast_tfv_std-std.inp	Second Abaqus/Standard import analysis.

Three-dimensional solid element tests:

solid_3d_elast_tfv_std.inp	First Abaqus/Standard analysis.
solid_3d_elast_tfv_std-std.inp	Second Abaqus/Standard import analysis.
solid_3d_elast_tfv_default_std-std.inp	Second Abaqus/Standard import analysis with default all section input.

Three-dimensional solid and structural element tests:

solid_strnb_elast_tfv_std.inp	First Abaqus/Standard analysis.
solid_strnb_elast_tfv_std-std.inp	Second Abaqus/Standard import analysis.

Transferring results with nodal temperature and field variables from an Abaqus/Explicit analysis to an Abaqus/Standard analysis

Elements tested

B31 C3D4 C3D6 C3D8 C3D8I C3D8R C3D10M
CPE3 CPE4R CPE6M CPS3 CPS4R CPS6M
M3D3 M3D4R S3R S4R SC8R

Problem description

The verification problems presented in this section test the transfer of results with nodal temperature and field variables from an Abaqus/Explicit analysis to an Abaqus/Standard analysis.

The model in each test consists of disjoint blocks of various types of elements. The blocks are fixed at one end and subject to concentrated loads at the other. The elements reference an elastic material for which the properties are given as tabulated data of the temperature and field variables 1 and 2. In Step 1 the steady-state response is computed in an Abaqus/Explicit analysis with given values of concentrated loads, nodal temperature, and nodal field variables. In Step 2 the model and the results at the end of Step 1 are imported to an Abaqus/Standard static analysis and the solution continues with the same load, nodal temperature, and field variable values used in Step-1. The nodal and elemental results of Step 2 in the import analysis are compared to those of Step 1 in the previous analysis.

Results and discussion

The nodal and elemental results in Step 2 of the import analysis are the same as those in Step 1 of the previous analysis. These tests verify that an analysis model with nodal temperature and field variables can be imported from an Abaqus/Explicit analysis into an Abaqus/Standard analysis.

Input files

Planar element tests:

solid_2d_elast_tfv_xpl.inp	First Abaqus/Explicit analysis.
solid_2d_elast_tfv_xpl-std.inp	Second Abaqus/Standard import analysis.

Three-dimensional solid element tests:

solid_3d_elast_tfv_xpl.inp	First Abaqus/Explicit analysis.
solid_3d_elast_tfv_xpl-std.inp	Second Abaqus/Standard import analysis.
solid_3d_elast_tfv_default_xpl-std.inp	Second Abaqus/Standard import analysis with default all section input.

Three-dimensional solid and structural element tests:

solid_str_elast_tfv_xpl.inp	First Abaqus/Explicit analysis.
solid_str_elast_tfv_xpl-std.inp	Second Abaqus/Standard import analysis.

Transferring results with nodal temperature and field variables from an Abaqus/Explicit analysis to an Abaqus/Explicit analysis

Elements tested

B31 C3D4 C3D6 C3D8 C3D8I C3D8R C3D10M
CPE3 CPE4R CPE6M CPS3 CPS4R CPS6M
M3D3 M3D4R SC8R S3R S4R

Problem description

The verification problems presented in this section test the transfer of results with nodal temperature and field variables from an Abaqus/Explicit analysis to an Abaqus/Explicit analysis.

The model in each test consists of disjoint blocks of various types of elements. The blocks are fixed at one end and subject to concentrated loads at the other. The elements reference an elastic material for which the properties are given as tabulated data of the temperature and field variables 1 and 2. In Step 1 the steady-state response is computed in an Abaqus/Explicit analysis with given values of concentrated loads, nodal temperature, and nodal field variables. In Step 2 the model and the results at the end of Step 1 are imported to an Abaqus/Explicit analysis and the solution continues with the same load, nodal temperature, and field variable values used in Step 1. The nodal and elemental results of Step 2 in the import analysis are compared to those of Step 1 in the previous analysis.

Results and discussion

The nodal and elemental results in Step 2 of the import analysis are the same as those in Step 1 of the previous analysis. These tests verify that an analysis model with nodal temperature and field variables can be imported from an Abaqus/Explicit analysis into an Abaqus/Explicit analysis.

Input files

Planar element tests:

solid_2d_elast_tfv_xpl.inp	First Abaqus/Explicit analysis.
solid_2d_elast_tfv_xpl-xpl.inp	Second Abaqus/Explicit import analysis.
solid_2d_elast_tfv_model_xpl-xpl.inp	Second Abaqus/Explicit import analysis with full model input.

Three-dimensional solid element tests:

solid_3d_elast_tfv_xpl.inp	First Abaqus/Explicit analysis.
solid_3d_elast_tfv_xpl-xpl.inp	Second Abaqus/Explicit import analysis.
solid_3d_elast_tfv_default_xpl-xpl.inp	Second Abaqus/Explicit import analysis with default all section input.
solid_3d_elast_tfv_model_xpl-xpl.inp	Second Abaqus/Explicit import analysis with full model input.
solid_3d_elast_tfv_model_default_xpl-xpl.inp	Second Abaqus/Explicit import analysis with full model and default all section input.

Three-dimensional solid and structural element tests:

solid_str_elast_tfv_xpl.inp	First Abaqus/Explicit analysis.
solid_str_elast_tfv_xpl-xpl.inp	Second Abaqus/Explicit import analysis.
solid_str_elast_tfv_default_xpl-xpl.inp	Second Abaqus/Explicit import analysis with default all section input.
solid_str_elast_tfv_model_xpl-xpl.inp	Second Abaqus/Explicit import analysis with full model input.

solid_str_elast_tfv_model_default_xpl-xpl.inp	Second Abaqus/Explicit import analysis with full model and default all section input.

Transferring results of an analysis model multiple times

Products: Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Transferring results of element sets multiple times from an Abaqus/Standard analysis to an Abaqus/Explicit analysis

Elements tested

B31 C3D4 C3D6 C3D8 C3D8R C3D10M T3D2
CPE3 CPE4R CPE6M CPS3 CPS4R CPS6M
M3D3 M3D4R SC8R S3R S4 S4R

Problem description

The verification problems presented in this section test the transfer of results of selected sets of elements multiple times from an Abaqus/Standard analysis to an Abaqus/Explicit analysis.

The model in each test consists of disjoint blocks of various types of elements subject to concentrated loads applied at one end of each block. The blocks are laid out in the X–Y plane, and all elements have elastic material properties. In Step 1 the static response is computed in an Abaqus/Standard analysis. In Step 2 the model and results at the end of Step 1 are imported into an Abaqus/Explicit analysis; the loading and boundary conditions from the previous analysis are maintained. In some tests the model is imported once and relocated by a 90° rotation of the model about the z-axis. In other tests the model is imported twice, first at the same location of the previous analysis and then at a new location that results from a 90° rotation of the model about the z-axis. The response of Step 2 in the import analysis is compared to that of Step 1 in the previous analysis.

Results and discussion

The nodal and elemental results in Step 2 of the import analysis are the same as those in Step 1 of the previous analysis for both elements that are placed at the original location and those that are placed in the new location during import. These tests verify that an analysis model can be imported from an Abaqus/Standard analysis into an Abaqus/Explicit analysis multiple times.

Input files

Planar elements tests:

import_multi_2d_elast_std.inp	First Abaqus/Standard analysis.
import_multi_2d_elast_std-xpl_state.inp	Second Abaqus/Explicit analysis: import all element sets twice, UPDATE=NO and STATE=YES.
import_multi_2d_elast_model_std-xpl_state.inp	Second Abaqus/Explicit analysis: import all element sets twice, UPDATE=NO and STATE=YES with full-model input.
import_multi_2d_elast_std-xpl_update.inp	Second Abaqus/Explicit analysis: import all element sets twice, UPDATE=YES and STATE=NO.

Solid elements tests:

import_solid_elast_std.inp	First Abaqus/Standard analysis.

import_solid_elast_std-xpl_state.inp	Second Abaqus/Explicit analysis: import all element sets once at the original location, UPDATE=NO and STATE=YES.
import_solid_elast_default_std-xpl_state.inp	Second Abaqus/Explicit analysis: import with default all section input at the original location, UPDATE=NO and STATE=YES.
import_solid_elast_std-xpl_update.inp	Second Abaqus/Explicit analysis: import all element sets once at the original location, UPDATE=YES and STATE=NO.
import_solid_elast_default_std-xpl_update.inp	Second Abaqus/Explicit analysis: import with default all section input at the original location, UPDATE=YES and STATE=NO.
import_solid_elast_std-xpl_rotate_state.inp	Second Abaqus/Explicit analysis: import all element sets once at the rotated location, UPDATE=NO and STATE=YES.
import_solid_elast_default_std-xpl_rotate_state.inp	Second Abaqus/Explicit analysis: import with default all section input at the rotated location, UPDATE=NO and STATE=YES.
import_solid_elast_std-xpl_rotate_update.inp	Second Abaqus/Explicit analysis: import all element sets once at the rotated location, UPDATE=YES and STATE=NO.
import_solid_elast_default_std-xpl_rotate_update.inp	Second Abaqus/Explicit analysis: import with default all section input at the rotated location, UPDATE=YES and STATE=NO.

Three-dimensional elements tests:

import_multi_3d_elast_std.inp	First Abaqus/Standard analysis.
import_multi_3d_elast_std-xpl_state.inp	Second Abaqus/Explicit analysis: import all element sets twice, UPDATE=NO and STATE=YES.
import_multi_3d_elast_model_std-xpl_state.inp	Second Abaqus/Explicit analysis: import all element sets twice, UPDATE=NO and STATE=YES with full-model input.
import_multi_3d_elast_std-xpl_update.inp	Second Abaqus/Explicit analysis: import all element sets twice, UPDATE=YES and STATE=NO.

Transferring results of element sets multiple times from an Abaqus/Standard analysis to an Abaqus/Standard analysis

Elements tested

C3D4 C3D6 C3D8 C3D8H C3D8I C3D8R C3D10M
CPE3 CPE4R CPE6M CPS3 CPS4R CPS6M
M3D3 M3D4R SC8R S3R S4R T3D2

Problem description

The verification problems presented in this section test the transfer of results of selected sets of elements multiple times from an Abaqus/Standard analysis to an Abaqus/Standard analysis.

The model in each test consists of disjoint blocks of various types of elements subject to concentrated loads applied at one end of each block. The blocks are laid out in the X–Y plane, and all elements have elastic material properties. In Step 1 the static response is computed in an Abaqus/Standard analysis. In Step 2 the model and results at the end of Step 1 are imported into an Abaqus/Standard analysis; the loading and boundary conditions from the previous analysis are maintained. In some tests the model is imported once and relocated by a 90° rotation of the model about the z-axis. In other tests the model is imported twice: first at the same location of the previous analysis, and then at a new location that results from a 90° rotation of the model about the z-axis. The static response of Step 2 in the import analysis is compared to that of Step 1 in the previous analysis.

Results and discussion

The nodal and elemental results in Step 2 of the import analysis are the same as those in Step 1 of the previous analysis for both elements that are placed at the original location and those that are placed in the new location during import. These tests verify that an analysis model can be imported from an Abaqus/Standard analysis into an Abaqus/Standard analysis multiple times.

Input files

Planar elements tests:

import_multi_2d_elast_s.inp	First Abaqus/Standard analysis.
import_multi_2d_elast_s2s_state.inp	Second Abaqus/Standard analysis: import all element sets twice, UPDATE=NO and STATE=YES.
import_multi_2d_elast_s2s_update.inp	Second Abaqus/Standard analysis: import all element sets twice, UPDATE=YES and STATE=NO.

Solid elements tests:

import_solid_elast_s.inp	First Abaqus/Standard analysis.
import_solid_elast_s2s_state.inp	Second Abaqus/Standard analysis: import all element sets once at the original location, UPDATE=NO and STATE=YES.
import_solid_elast_s2s_update.inp	Second Abaqus/Standard analysis: import all element sets once at the original location, UPDATE=YES and STATE=NO.
import_solid_elast_s2s_rotate_state.inp	Second Abaqus/Standard analysis: import all element sets once at the rotated location, UPDATE=NO and STATE=YES.
import_solid_elast_s2s_rotate_update.inp	Second Abaqus/Standard analysis: import all element sets once at the rotated location, UPDATE=YES and STATE=NO.

Three-dimensional elements tests:

import_multi_3d_elast_s.inp	First Abaqus/Standard analysis.
import_multi_3d_elast_s2s_state.inp	Second Abaqus/Standard analysis: import all element sets twice, UPDATE=NO and STATE=YES.
import_multi_3d_elast_s2s_update.inp	Second Abaqus/Standard analysis: import all element sets twice, UPDATE=YES and STATE=NO.

Transferring results of element sets multiple times from an Abaqus/Explicit analysis to an Abaqus/Explicit analysis

Elements tested

B31 C3D4 C3D6 C3D8 C3D8R C3D10M T3D2
CPE3 CPE4R CPE6M CPS3 CPS4R CPS6M
M3D3 M3D4R SC8R S3R S4 S4R

Problem description

The verification problems presented in this section test the transfer of results of selected sets of elements multiple times from one Abaqus/Explicit analysis to another Abaqus/Explicit analysis.

The model in each test consists of disjoint blocks of various types of elements subject to concentrated loads applied at one end of each block. The blocks are laid out in the X–Y plane, and all elements have elastic material properties. In Step 1 the steady-state response is computed in an Abaqus/Explicit analysis. In Step 2 the model

and results at the end of Step 1 are imported into another Abaqus/Explicit analysis; the loading and boundary conditions from the previous analysis are maintained. In some tests the model is imported once and relocated by a 90° rotation of the model about the z-axis. In other tests the model is imported twice, first at the same location of the previous analysis, and then at a new location that results from a 90° rotation of the model about the z-axis. The response of Step 2 in the import analysis is compared to that of Step 1 in the previous analysis.

Results and discussion

The nodal and elemental results in Step 2 of the import analysis are the same those in Step 1 of the previous analysis for both elements that are placed at the original location and those that are placed in the new location during import. These tests verify that an analysis model can be imported from one Abaqus/Explicit to another Abaqus/Explicit multiple times.

Input files

Planar elements tests:

import_multi_2d_elast_xpl.inp	First Abaqus/Explicit analysis.
import_multi_2d_elast_xpl-xpl_state.inp	Second Abaqus/Explicit analysis: import all element sets twice, UPDATE=NO and STATE=YES.
import_multi_2d_elast_model_xpl-xpl_state.inp	Second Abaqus/Explicit analysis: import all element sets twice, UPDATE=NO and STATE=YES with full-model input.
import_multi_2d_elast_xpl-xpl_update.inp	Second Abaqus/Explicit analysis: import all element sets twice, UPDATE=YES and STATE=NO.

Solid elements tests:

import_solid_elast_xpl.inp	First Abaqus/Explicit analysis.
import_solid_elast_xpl-xpl_state.inp	Second Abaqus/Explicit analysis: import all element sets once at the original location, UPDATE=NO and STATE=YES.
import_solid_elast_default_xpl-xpl_state.inp	Second Abaqus/Explicit analysis: import with default all section input at the original location, UPDATE=NO and STATE=YES.
import_solid_elast_xpl-xpl_update.inp	Second Abaqus/Explicit analysis: import all element sets once at the original location, UPDATE=YES and STATE=NO.
import_solid_elast_default_xpl-xpl_update.inp	Second Abaqus/Explicit analysis: import with default all section input at the original location, UPDATE=YES and STATE=NO.
import_solid_elast_xpl-xpl_rotate_state.inp	Second Abaqus/Explicit analysis: import all element sets once at the rotated location, UPDATE=NO and STATE=YES.
import_solid_elast_default_xpl-xpl_rotate_state.inp	Second Abaqus/Explicit analysis: import with default all section input at the rotated location, UPDATE=NO and STATE=YES.
import_solid_elast_xpl-xpl_rotate_update.inp	Second Abaqus/Explicit analysis: import all element sets once at the rotated location, UPDATE=YES and STATE=NO.
import_solid_elast_default_xpl-xpl_rotate_update.inp	Second Abaqus/Explicit analysis: import with default all section input at the rotated location, UPDATE=YES and STATE=NO.

Three-dimensional elements tests:

import_multi_3d_elast_xpl.inp	First Abaqus/Explicit analysis.
import_multi_3d_elast_xpl-xpl_state.inp	Second Abaqus/Explicit analysis: import all element sets twice, UPDATE=NO and STATE=YES.
import_multi_3d_elast_model_xpl-xpl_state.inp	Second Abaqus/Explicit analysis: import all element sets twice, UPDATE=NO and STATE=YES with full-model input.

import_multi_3d_elast_xpl-xpl_update.inp	Second Abaqus/Explicit analysis: import all element sets twice, UPDATE=YES and STATE=NO.

Transferring results of a quarter circular plate modeled with a part instance four times from an Abaqus/Standard analysis to an Abaqus/Explicit analysis of the whole plate

Elements tested

S4R

Problem description

The quarter circular plate model is assembled from a part consisting of elastic-plastic S4R shell elements. Uniform pressure loading is applied on the top surface of the plate. The circular edge is clamped, and symmetry boundary conditions are specified along the cut edges. In Step 1 an Abaqus/Standard analysis is performed to compute the static response. In Step 2 the quarter model and results at the end of Step 1 are imported into an Abaqus/Explicit analysis four times: first at the original location, then relocated with rotations of 90°, 180°, and −90°. The result is a model for the whole circular plate with the four quarter circles disconnected along the original cut edges. Tie constraints are defined to connect these edges so that the response of the model corresponds to that of a continuous plate. The Abaqus/Explicit import analysis is performed with the same loading as in the previous static Abaqus/Standard analysis, and the clamped boundary conditions on the circular boundary are maintained.

Results and discussion

In the Abaqus/Explicit import analysis of the whole plate, the steady-state response is the same as that of the quarter plate model computed in the previous static Abaqus/Standard analysis. This result verifies that sets of S4R elements, including model and states, can be transferred multiple times from an Abaqus/Standard analysis to an Abaqus/Explicit analysis.

Input files

cplate_assem_std.inp	First Abaqus/Standard analysis.
cplate_assem_std-xpl_x4.inp	Second Abaqus/Explicit analysis: import S4R element set four times, UPDATE=NO and STATE=YES.

Transferring results of a quarter circular plate modeled with a part instance four times from an Abaqus/Standard analysis to an Abaqus/Standard analysis of the whole plate

Elements tested

S4R

Problem description

The quarter circular plate model is assembled from a part consisting of elastic-plastic S4R shell elements. Uniform pressure loading is applied on the top surface of the plate. The circular edge is clamped, and symmetry boundary conditions are specified along the cut edges. In Step 1 an Abaqus/Standard analysis is performed to compute the static response. In Step 2 the quarter model and results at the end of Step 1 are imported into an Abaqus/Standard analysis four times: first at the original location, then relocated with rotations of 90°, 180°, and −90°. The result is a model for the whole circular plate with the four quarter circles disconnected along the original cut edges. Tie constraints are defined to connect these edges so that the response of the model corresponds to that of a continuous plate. The Abaqus/Standard import static analysis is performed with the same loading as in the previous static Abaqus/Standard analysis, and the clamped boundary conditions on the circular boundary are maintained.

Results and discussion

In the Abaqus/Standard import analysis of the whole plate, the static response is the same as that of the quarter plate model computed in the previous static Abaqus/Standard analysis. This result verifies that sets of S4R elements, including model and states, can be transferred multiple times from an Abaqus/Standard analysis to an Abaqus/Standard analysis.

Input files

cplate_assem_std.inp	First Abaqus/Standard analysis.
cplate_assem_std-std_x4.inp	Second Abaqus/Standard analysis: import S4R element set four times, UPDATE=NO and STATE=YES.

Transferring results of a quarter circular plate modeled with a part instance four times from an Abaqus/Explicit analysis to an Abaqus/Standard analysis of the whole plate

Elements tested

S4R

Problem description

The quarter circular plate model is assembled from a part consisting of elastic-plastic S4R shell elements. Uniform pressure loading is applied on the top surface of the plate. The circular edge is clamped, and symmetry boundary conditions are specified along the cut edges. In Step 1 an Abaqus/Explicit analysis is performed to compute the steady-state response. In Step 2 the quarter model and results at the end of Step 1 are imported into an Abaqus/Standard static analysis four times: first at the original location, and then relocated with rotations of 90°, 180°, and −90°. The result is a model for the whole circular plate with the four quarter circles disconnected along the original cut edges. Tie constraints are defined to connect these edges so that the response of the model corresponds to that of a continuous plate. The Abaqus/Standard static import analysis is performed with the same loading as in the previous Abaqus/Explicit analysis, and the clamped boundary conditions on the circular boundary are maintained.

Results and discussion

In the Abaqus/Standard import analysis of the whole plate, the static response is the same as that of the quarter plate model computed in the previous steady-state Abaqus/Explicit analysis. This result verifies that sets of S4R elements, including model and states, can be transferred multiple times from one Abaqus/Explicit analysis to an Abaqus/Standard analysis.

Input files

cplate_assem_xpl.inp	First Abaqus/Explicit analysis.
cplate_assem_xpl-std_x4.inp	Second Abaqus/Standard analysis: import the S4R element set four times, UPDATE=NO and STATE=YES.

Transferring results of a quarter circular plate modeled with a part instance four times from an Abaqus/Explicit analysis to an Abaqus/Explicit analysis of the whole plate

Elements tested

S4R

Problem description

The quarter circular plate model is assembled from a part consisting of elastic-plastic S4R shell elements. Uniform pressure loading is applied on the top surface of the plate. The circular edge is clamped, and symmetry boundary conditions are specified along the cut edges. In Step 1 an Abaqus/Explicit analysis is performed to compute the steady-state response. In Step 2 the quarter model and results at the end of Step 1 are imported into another Abaqus/Explicit analysis four times: first at the original location, and then relocated with rotations of 90°, 180°, and −90°. The result is a model for the whole circular plate with the four quarter circles disconnected along the original cut edges. Tie constraints are defined to connect these edges so that the response of the model corresponds to that of a continuous plate. The Abaqus/Explicit import analysis is performed with the same loading as in the previous Abaqus/Explicit analysis, and the clamped boundary conditions on the circular boundary are maintained.

Results and discussion

In the Abaqus/Explicit import analysis of the whole plate, the steady-state response is the same as that of the quarter plate model computed in the previous steady-state Abaqus/Explicit analysis. This result verifies that sets of S4R elements, including model and states, can be transferred multiple times from one Abaqus/Explicit analysis to another Abaqus/Explicit analysis.

Input files

cplate_assem_xpl.inp	First Abaqus/Explicit analysis.
cplate_assem_xpl-xpl_x4.inp	Second Abaqus/Explicit analysis: import the S4R element set four times, UPDATE=NO and STATE=YES.

Transferring results of the forming of a quarter conical shell modeled with element sets four times from an Abaqus/Standard analysis to an Abaqus/Standard analysis of the whole shell

Elements tested

C3D8R S3 S4R

Problem description

The quarter model of the forming of a conical aluminum shell consists of elastic S3 and S4R shell elements pressed against an outer mold modeled as a rigid body of C3D8R elements. Uniform pressure loading is applied on the shell surface. A contact pair is defined for the surfaces of the shell and the mold, and surface smoothing of a variety of geometries is specified between the contacting surfaces. The reference node of the rigid mold is fixed, and symmetry boundary conditions are specified along the cut edges. In Step 1 an Abaqus/Standard analysis is performed to compute the static response. In Step 2 the quarter model and results at the end of Step 1 are imported into an Abaqus/Standard analysis four times: first at the original location, then relocated with rotations of −90°, −180°, and −270°. The result is a model for the whole conical shell with the four quarter shells that are disconnected along the original cut edges. For verification purpose these edges are not tied. The Abaqus/Standard import static analysis is performed with the same loading as in the previous static Abaqus/Standard analysis, and the fixed boundary conditions on the reference nodes are maintained.

Results and discussion

In the Abaqus/Standard import analysis of the whole shell, the static response is the same as that of the quarter shell model computed in the previous static Abaqus/Standard analysis. This result verifies that the model and states of sets of C3D8R, S3, and S4R elements—together with the contact pair and surface smoothing definitions—can be transferred multiple times from an Abaqus/Standard analysis to an Abaqus/Standard analysis.

Input files

surfsmooth_import_s2s_j1.inp	First Abaqus/Standard analysis.
surfsmooth_import_s2s_j2_x4.inp	Second Abaqus/Standard analysis: import quarter model four times, UPDATE=NO and STATE=YES.

Transferring results from multiple previous analyses

Products: Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Transferring results of element sets from multiple Abaqus/Standard analyses to an Abaqus/Explicit analysis

Elements tested

C3D8R S4R

Problem description

The verification problems presented in this section test the transfer of results of selected element sets from two Abaqus/Standard analyses to an Abaqus/Explicit analysis.

The first Abaqus/Standard analysis consists of a quarter circular plate in the X–Y plane modeled with elastic-plastic S4R shell elements. The circular edge is clamped, and symmetry boundary conditions are specified along the cut edges. Uniform pressure loading is applied on the top of the plate. In Step 1 an Abaqus/Standard analysis runs to compute the static response.

The second Abaqus/Standard analysis consists of a partially rigid rotating shaft modeled with elastic C3D8R solid elements. One of the solid elements is deformable, and the rest are rigid bodies whose reference nodes all lie on the longitudinal axis of the shaft. Different rotations about the axis of the shaft are specified on the reference nodes of the rigid body. In Step 1 an Abaqus/Standard analysis runs to compute the static response.

In Step 2 the model and results of the two Abaqus/Standard analyses are imported into an Abaqus/Explicit analysis. The solid element model is imported once, and the shell element model is imported four times: first at the original location and then relocated with rotations of 90°, 180°, and −90° about the z-axis. The result is a model for the whole circular plate formed by the four quarter circles. The Abaqus/Explicit import analysis is performed with the same loading on the circular plate as in the previous static Abaqus/Standard analysis, and the rigid solid elements are given additional rotations. The clamped and symmetry boundary conditions are maintained for the circular plate model.

Results and discussion

The nodal and elemental results of the circular plate in Step 2 of the import analysis are the same as those in Step 1 of the previous analysis, and the rigid shaft elements are rotated further to the correct specified locations. These tests verify that analysis models consisting of C3D8R and S4R elements can be imported from multiple Abaqus/Standard analyses into an Abaqus/Explicit analysis.

Input files

cplate_std.inp	Quarter circular plate, Abaqus/Standard analysis.
rshaft_std.inp	Partially rigid shaft, Abaqus/Standard analysis.
cplate_rshaft_std-xpl_x4.inp	Multiple library Abaqus/Explicit import analysis with circular plate and partially rigid shaft as old jobs 1 and 2, respectively, and using update options UPDATE=NO and STATE=YES.

cplate_rshaft_std-xpl_x4_update.inp	Multiple library Abaqus/Explicit import analysis with circular plate and partially rigid shaft as old jobs 1 and 2, respectively, and using update options UPDATE=YES and STATE=NO.
rshaft_cplate_std-xpl_x4.inp	Multiple library Abaqus/Explicit import analysis with partially rigid shaft and circular plate as old jobs 1 and 2, respectively, and using update options UPDATE=NO and STATE=YES.

Transferring results of element sets from multiple Abaqus/Explicit analyses to an Abaqus/Explicit analysis

Elements tested

C3D8R S4R

Problem description

The verification problems presented in this section test the transfer of results of selected element sets from two Abaqus/Explicit analyses to another Abaqus/Explicit analysis.

The first Abaqus/Explicit analysis consists of a quarter circular plate X–Y plane modeled with elastic-plastic S4R shell elements. The circular edge is clamped, and symmetry boundary conditions are specified along the cut edges. Uniform pressure loading is applied on the top of the plate. In Step 1 an Abaqus/Explicit analysis runs to compute the steady-state response.

The second Abaqus/Explicit analysis consists of a partially rigid rotating shaft modeled with elastic C3D8R solid elements. One of the solid elements is deformable, and the rest are rigid bodies whose reference nodes all lie on the longitudinal axis of the shaft. Different rotations about the axis of the shaft are specified on the reference nodes of the rigid body. In Step 1 an Abaqus/Explicit analysis runs to compute the steady-state response.

In Step 2 the model and results of the two Abaqus/Explicit analyses are imported into an Abaqus/Explicit analysis. The solid element model is imported once, and the shell element model is imported four times: first at the original location and then relocated with rotations of 90°, 180°, and −90° about the z-axis. The result is a model for the whole circular plate formed by the four quarter circles. The Abaqus/Explicit import analysis runs with the same loading on the circular plate as in the previous steady-state Abaqus/Explicit analysis, and the rigid solid elements are given additional rotations. The clamped and symmetry boundary conditions are maintained for the circular plate model.

Results and discussion

The nodal and elemental results of the circular plate in Step 2 of the import analysis are the same as those in Step 1 of the previous analysis, and the rigid shaft elements are rotated further to the correct specified locations. These tests verify that analysis models consisting of C3D8R and S4R elements can be imported from multiple Abaqus/Explicit analyses into another Abaqus/Explicit analysis.

Input files

cplate_xpl.inp	Quarter circular plate, Abaqus/Explicit analysis.
rshaft_xpl.inp	Partially rigid shaft, Abaqus/Explicit analysis.
cplate_rshaft_xpl-xpl_x4.inp	Multiple library Abaqus/Explicit import analysis with circular plate and partially rigid shaft as old jobs 1 and 2, respectively, and using update options UPDATE=NO and STATE=YES.
cplate_rshaft_xpl-xpl_x4_update.inp	Multiple library Abaqus/Explicit import analysis with circular plate and partially rigid shaft as old jobs 1 and 2, respectively, and using update options UPDATE=YES and STATE=NO.

rshaft_cplate_xpl-xpl_x4.inp	Multiple library Abaqus/Explicit import analysis with partially rigid shaft and circular plate as old jobs 1 and 2, respectively, and using update options UPDATE=NO and STATE=YES.

Transferring results with nodal temperature and field variables from multiple Abaqus/Standard analyses to an Abaqus/Explicit analysis

Elements tested

C3D8R S4R

Problem description

The verification problem tests the transfer of results with nodal temperature and field variables from four Abaqus/Standard analyses to an Abaqus/Explicit analysis.

The first and second Abaqus/Standard analysis models each consist of a rectangular column modeled with C3D8R solid elements. The column is fixed at one end and subject to compression at the other end. In the first analysis the column is modeled with an elastic material with constant properties. In the second analysis the elastic properties are dependent on the temperature and two field variables. In Step 1 the static response is computed for both analyses for given values of the compression and the temperature and field variables.

The third and fourth Abaqus/Standard analysis models each consist of a rectangular plate modeled with S4R shell elements. In the third analysis the plate is modeled with an elastic material with properties that are dependent on the temperature and two field variables, and seven temperature points per shell node are specified. In the fourth analysis the elastic properties are dependent on three field variables, and five temperature points per shell node are specified. In Step 1 the static response is computed for both analyses for given values of the compression and the temperature and field variables.

In Step 2 the model and results of the four Abaqus/Standard analyses are imported into an Abaqus/Explicit analysis and each model is repositioned using translation and rotation. In addition, a new rectangular column modeled with C3D8R solid elements is added to the import analysis. The new column is fixed at one end and subject to compression at the other end. It is modeled with elastic properties that are dependent on the temperature and one field variable.

In the Abaqus/Explicit import analysis, the same end conditions and temperature and field variables values used in Step 1 are applied to the imported models. The initial response of the imported models is compared with the Step 1 results of the corresponding previous analyses.

Results and discussion

The initial nodal and elemental results of each of the four imported models in Step 2 are the same as those in Step 1 of the corresponding previous analysis. This test verifies that analysis models consisting of C3D8R and S4R elements with nodal temperature and field variables can be imported from multiple Abaqus/Standard analyses into an Abaqus/Explicit analysis.

Input files

c3d8r_tfv0_std.inp	Column modeled with solid elements, first Abaqus/Standard analysis.
c3d8r_tfv2_std.inp	Column modeled with solid elements, second Abaqus/Standard analysis.
s4r_htfv2_t7_std.inp	Plate modeled with shell elements, third Abaqus/Standard analysis.
s4r_htfv3_t5_std.inp	Plate modeled with shell elements, fourth Abaqus/Standard analysis.
mlib_c3d8r_02_s4r_23_add_1_std-xpl.inp	Multiple library Abaqus/Explicit import analysis with the above four Abaqus/Standard analyses.

mlib_c3d8r_02_s4r_23_add_1_model_std-xpl.inp	Multiple library Abaqus/Explicit import analysis with the above four Abaqus/Standard analyses with full-model input.

Transferring results with nodal temperature and field variables from multiple Abaqus/Explicit analyses to an Abaqus/Explicit analysis

Elements tested

C3D8R S4R

Problem description

The verification problem tests the transfer of results with nodal temperature and field variables from four Abaqus/Explicit analyses to an Abaqus/Explicit analysis.

The first and second Abaqus/Explicit analysis models each consist of a rectangular column modeled with C3D8R solid elements. The column is fixed at one end and subject to compression at the other end. In the first analysis, the column is modeled with an elastic material with constant properties. In the second analysis, the elastic properties are dependent on the temperature and two field variables. In Step 1 the dynamic response is computed for both analyses for given values of the compression and the temperature and field variables.

The third and fourth Abaqus/Explicit analysis models each consist of a rectangular plate modeled with S4R shell elements. In the third analysis, the plate is modeled with an elastic material with properties that are dependent on the temperature and two field variables, and seven temperature points per shell node are specified. In the fourth analysis, the elastic properties are dependent on three field variables, and five temperature points per shell node are specified. In Step 1 the dynamic response is computed for both analyses for given values of the compression and the temperature and field variables.

In Step 2 the model and results of the four Abaqus/Explicit analyses are imported into an Abaqus/Explicit analysis, and each model is repositioned using translation and rotation. In addition, a new rectangular column modeled with C3D8R solid elements is added to the import analysis. The new column is fixed at one end and subject to compression at the other end. It is modeled with elastic properties that are dependent on the temperature and one field variable.

In the Abaqus/Explicit import analysis, the same end conditions and temperature and field variables values as used in Step 1 are applied to the imported models. The initial response of the imported models is compared with the Step 1 results of the corresponding previous analyses.

Results and discussion

The initial nodal and elemental results of each of the four imported models in Step 2 are the same as those in Step 1 of the corresponding previous analysis. This test verifies that analysis models consisting of C3D8R and S4R elements with nodal temperature and field variables can be imported from multiple Abaqus/Explicit analyses into an Abaqus/Explicit analysis.

Input files

c3d8r_tfv0_xpl.inp	Column modeled with solid elements, first Abaqus/Explicit analysis.
c3d8r_tfv2_xpl.inp	Column modeled with solid elements, second Abaqus/Explicit analysis.
s4r_htfv2_t7_xpl.inp	Plate modeled with shell elements, third Abaqus/Explicit analysis.
s4r_htfv3_t5_xpl.inp	Plate modeled with shell elements, fourth Abaqus/Explicit analysis.
mlib_c3d8r_02_s4r_23_add_1_xpl-xpl.inp	Multiple library Abaqus/Explicit import analysis with the above four Abaqus/Explicit analyses.

mlib_c3d8r_02_s4r_23_add_1_model_xpl-xpl.inp	Multiple library Abaqus/Explicit import analysis with the above four Abaqus/Explicit analyses with full-model input.

Meshed beam cross-sections

In this section:

- *About meshed beam cross-sections*
- *Meshing and analyzing a two-dimensional model of a beam cross-section*
- *Using generated cross-section properties in a beam analysis*

About meshed beam cross-sections

Overview

The meshed beam cross-section capability allows for the description of a beam cross-section that is geometrically complex or composed of more than one material.
The meshed cross-section modeling approach is intended for structures that are expected to respond like beams but do not permit the use of a predefined cross-section shape.

To use meshed beam cross-sections in a beam analysis, the beam cross-section is first meshed with two-dimensional warping elements. The meshed cross-section is used to numerically integrate the beam stiffness and inertia properties and to calculate the warping functions in Abaqus/Standard. The two-dimensional Abaqus/Standard analysis writes the cross-sectional properties to an input-file-ready text file called *jobname*`.bsp`. This file is used to define the appropriate section stiffness and inertia data for a subsequent Abaqus/Standard or Abaqus/Explicit beam element analysis. The cross-section is pre-integrated and remains elastic throughout the analysis (beam general section). The generated beam cross-section properties include the axial, bending, torsional, and transverse shear stiffness; mass, rotary inertia, and damping properties; and the location of the centroid and shear center. In addition, the equivalent beam cross-section properties include information on stress recovery, such as the warping function and its derivatives. Once the beam element analysis is complete, the Visualization module of Abaqus/CAE can be used to visualize the results at preselected points along the beam length or to examine detailed stress and strain results in the two-dimensional meshed cross-section.

The verification tests that follow are divided into two sections. The first section contains analyses in which the cross-section properties for two-dimensional models of meshed cross-sections are obtained. The cross-section shapes include the standard beam sections that are available for use with beam elements, such as I-sections or rectangular sections, and nonstandard beam sections, such as C-sections and airfoil sections. The second section verifies the results obtained for beam analyses using an arbitrarily shaped solid general beam cross section meshed with warping elements by comparing them with the results obtained using a linear generalized cross section for a number of different procedure types.

Meshing and analyzing a two-dimensional model of a beam cross-section

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

WARP2D3 WARP2D4 WARPF2D3 WARPF2D4 WARPF2D6 WARPF2D8

Features tested

The special-purpose two-dimensional elements WARP2D3 (3-node triangular), WARP2D4 (4-node quadrilateral), WARPF2D3 (3-node triangular), WARPF2D4 (4-node quadrilateral), WARPF2D6 (6-node triangular), and WARPF2D8 (8-node quadrilateral) are used to create two-dimensional beam cross-section models. The beam section generation procedure is used to numerically calculate the geometric, stiffness, and inertia properties of the section, including warping functions and the locations of the elastic and shear center (see *Meshed beam cross-sections*). The calculated properties are written to the *jobname*`.bsp` text file.

Problem description

Model: Several cross-section shapes are considered. Two-dimensional finite element models of an I-section, an I-section with nodal offset, a rectangular section, a pipe section with a cut, a C-section, an airfoil section (see *Figure 1*), and a wind turbine rotor blade section (see *Figure 2*) are included.

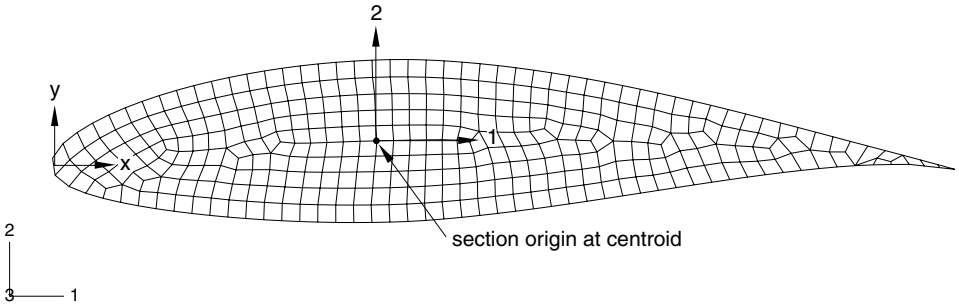
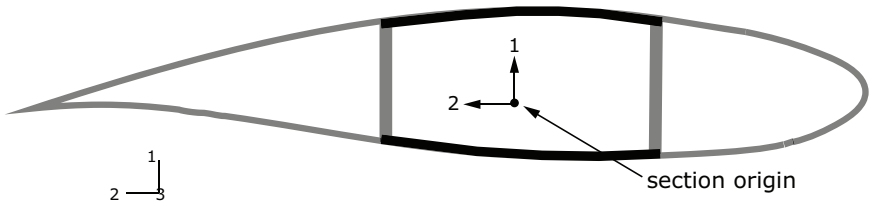


Figure 1: Airfoil cross-section.



Geometry from the National Renewable Energy Laboratory, a national laboratory of the U.S. Department of Energy.

Figure 2: Wind turbine rotor blade section.

Mesh: There are two sets of element types that cannot be mixed in the same mesh. Elements of type WARP2D3 and WARP2D4 have one degree of freedom per node for the out-of-plane warping displacement for a unit twist of an associated beam element. Elements of type WARPF2D3, WARPF2D4, WARPF2D6, and WARPF2D8 have three degrees of freedom per node for the in-plane and out-of-plane warping displacements for a unit beam element section force or section moment. The cross-sections use only one set of element types.

Material: Only elastic materials, using either linear isotropic or orthotropic shear behavior, can be used for two-dimensional models with 1-DOF warping elements. For two-dimensional models with 3-DOF warping elements, all linear elastic materials available for three-dimensional solid elements can be used (isotropic, orthotropic, and fully anisotropic elasticity).

Boundary conditions: Boundary conditions are not meaningful when generating beam section properties and are ignored.

Loading: Loads are not meaningful when generating beam section properties and are ignored.

Results and discussion

For cross-sections with 1-DOF warping elements and 3-DOF warping elements and isotropic material, the beam cross-section properties for each of the meshed cross-sections are written to the *jobname*`.bsp` text file if the directional coupling in the beam element response to stretching, shearing, bending, and twisting can be fully characterized by the elastic center and shear center locations.

In all cases, the beam cross-section properties are reported in the *jobname*`.dat` file.

For cross-sections with 3-DOF warping elements, the properties and the warping functions are also written to a *jobname*`.fbi` binary file.

The integrated values of the properties for the meshed beam cross-sections are compared to the analytical solutions or solutions generated for a section from the predefined library. The warping function shapes of the two-dimensional cross-sections compare well with the solutions for the solid element models of the beam subjected to a unit twist in the case of cross-sections with 1-DOF warping elements or subjected to the six unit section forces and moments in the case of cross-sections with 3-DOF warping elements.

Input files

meshedsect_airfoil.inp	Meshed cross-section of an airfoil.
meshedsect_c.inp	Meshed C cross-section.
meshedsect_c_warpf2d4.inp	Meshed C cross-section with 3-DOF warping elements.
meshedsect_c_warpf2d4_bent.inp	Meshed C cross-section with 3-DOF warping elements, considering prebending of the beam.
meshedsect_c_warpf2d4_inclined.inp	Meshed C cross-section with 3-DOF warping elements, considering an inclination of the cross-section plane with respect to the beam centerline.
meshedsect_c_warpf2d4_orthotropic.inp	Meshed C cross-section with 3-DOF warping elements and orthotropic material.

meshedsect_c_warpf2d4_sc.inp	Meshed C cross-section with 3-DOF warping elements and the reference point at the shear center.
meshedsect_c_warpf2d4_twisted.inp	Meshed C cross-section with 3-DOF warping elements, considering pretwist of the beam.
meshedsect_cutcircle.inp	Meshed pipe cross-section with cut.
meshedsect_i_iso.inp	Meshed I cross-section with isotropic material.
meshedsect_i_ortho.inp	Meshed I cross-section with orthotropic material.
meshedsect_i_orthoiso.inp	Meshed I cross-section with TRACTION material.
meshedsect_i_twomat.inp	Meshed I cross-section with two different materials.
meshedsect_i_offset.inp	Meshed I cross-section with nodal offset.
meshedsect_rectangle.inp	Meshed rectangular cross-section.
meshedsect_rotorblade_nrel5mw_at50m.inp	Meshed rotor blade cross-section, loosely based on the NREL 5MW reference wind turbine at a distance of 50 m from the blade root.

Using generated cross-section properties in a beam analysis

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

B21 B22 B31 B32

Features tested

The cross-section properties generated and stored in the *jobname*`.bsp` text files from the previous section, *Meshing and analyzing a two-dimensional model of a beam cross-section*, are used in beam analyses to define stiffness and inertia properties for beam elements. The arbitrarily shaped solid general beam cross section meshed with warping elements option is used to assign the precalculated stiffness and inertia properties to the beam elements. The results obtained for the meshed section beams are verified by comparing them with results obtained for beams using a linear generalized cross section that are assigned stiffness and inertia properties identical to those of the meshed beams. Dynamic analyses are performed in both Abaqus/Standard and Abaqus/Explicit. Static and frequency extraction analyses are also performed in Abaqus/Standard. The use of beams with additional inertia and additional fluid inertia to modify the inertia properties defined for the meshed beams in the *jobname*`.bsp` text files is also tested.

Problem description

Model: The model comprises a single cantilevered beam subjected to a concentrated load at its tip. The load is applied as a step load resulting in significant dynamic motion of the beam.

Results and discussion

The beam responses obtained using the arbitrarily shaped solid general beam cross section meshed with warping elements option are identical to the beam responses obtained using a linear generalized cross section. For the same model, the results from a dynamic analysis in Abaqus/Standard agree well with the results from a dynamic analysis in Abaqus/Explicit.

Input files

xbgs_meshedcsect_std.inp	Meshed beams with C-section; comparison of meshed and general section beams; dynamic analysis in Abaqus/Standard.
xbgs_meshedcsect_warpf2d4_std.inp	Meshed beams with C-section; 3-DOF warping elements; comparison of meshed and general section beams; dynamic analysis in Abaqus/Standard.
xbgs_meshedcsect_xpl.inp	Meshed beams with C-section; comparison of meshed and general section beams; dynamic analysis in Abaqus/Explicit.
xbgs_meshedisectiso_std.inp	Meshed beams with I-section; comparison of meshed and general section beams; dynamic analysis in Abaqus/Standard.
xbgs_meshedisectiso_xpl.inp	Meshed beams with I-section; comparison of meshed and general section beams; dynamic analysis in Abaqus/Explicit.

xbgs_meshedisectortho_std.inp	Meshed beams with I-section; composite elastic material; comparison of meshed and general section beams; dynamic analysis in Abaqus/Standard.
xbgs_meshedisectortho_xpl.inp	Meshed beams with I-section; composite elastic material; comparison of meshed and general section beams; dynamic analysis in Abaqus/Explicit.
xbgs_meshedisectoffset_std.inp	Meshed beams with I-section; centroid offset from reference node; comparison of meshed and general section beams; dynamic analysis in Abaqus/Standard.
xbgs_meshedisectoffset_xpl.inp	Meshed beams with I-section; centroid offset from reference node; comparison of meshed and general section beams; dynamic analysis in Abaqus/Explicit.
xbgs2d_meshedisectortho_std.inp	Two-dimensional meshed beams with I-section; composite elastic material; comparison of meshed and general section beams; dynamic analysis in Abaqus/Standard.
xbgs2d_meshedisectortho_xpl.inp	Two-dimensional meshed beams with I-section; composite elastic material; comparison of meshed and general section beams; dynamic analysis in Abaqus/Explicit.
xbgs_meshedrectsect_freq.inp	Meshed beams with rectangular section; comparison of meshed and general section beams; frequency extraction in Abaqus/Standard.
b31_i_bai_meshed.inp	Beam added inertia with meshed beams in Abaqus/Standard.
xbgs_meshedbai_xpl.inp	Beam added inertia with meshed beams in Abaqus/Explicit.
xbgs_meshedbfi_xpl.inp	Beam fluid inertia with meshed beams in Abaqus/Explicit.

Co-simulation

In this section:

- *Fluid-structure interaction of a cantilever beam inside a channel*
- *Abaqus/Standard to Abaqus/Explicit co-simulation*

Fluid-structure interaction of a cantilever beam inside a channel

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Unidirectional solution transfer between Abaqus/Standard and FLUENT

Elements tested

CPS4I

Features tested

- Unidirectional coupling between Abaqus/Standard and FLUENT; and
- transfer of normal surface pressure and concentrated forces.

Problem description

This verification problem illustrates the co-simulation feature used to couple Abaqus/Standard with FLUENT to perform a fluid-structure interaction (FSI) simulation. The problem consists of a slender cantilever beam placed inside a channel with steady, incompressible, laminar flow. For this case a unidirectional coupling is considered in which fluid pressure along the cantilever beam is computed by FLUENT and is imported into Abaqus. The problem is simple such that comparison between the numerical and analytical results can be made.

Model: The model consists of a slender cantilever beam inside a channel, as illustrated in *Figure 1*. The beam length is 1 m, and the thickness is 0.01 m. The depth is considered sufficiently large so that end effects can be neglected and the flow can be considered independent of the z position. A 0.1 m slice of the beam and channel is chosen for this model. The FLUENT model contains two fluid domains that are distinct at one end and merge at the opposite end of the beam: the top channel height is 0.02 m, and the bottom channel height is 0.04 m. The channel cross-section is uniform along the beam.

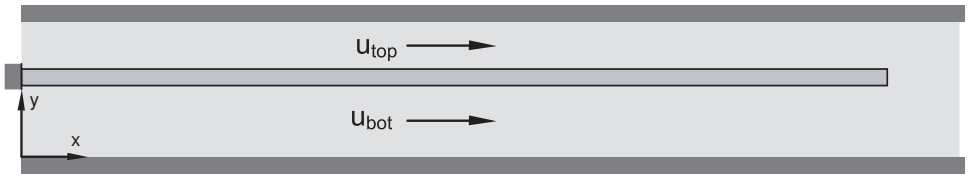


Figure 1: Schematic of channel flow problem.

Mesh: A two-dimensional model is used. The mesh consists of incompatible mode plane stress elements: 100 elements along the length, and 4 elements stacked in the thickness direction. No mesh parameter studies were performed on the structural mesh. The fluid-structure interface is defined through a surface definition.

The fluid mesh consists of 200 quadrilateral cells along the channel length and 8 cells and 16 cells stacked in the top and bottom channels, respectively. Quadrilateral fluid cells were used since these generally provide better pressure results than triangular fluid cells at the faces.

Material: The structural model uses linear elastic properties with Young's modulus of 1.09 GPa and a Poisson's ratio of 0.3.

The fluid model assumes incompressible flow with a fluid density of 1000 kg/m^3 and a dynamic viscosity of 0.001 kg/ms.

Boundary conditions: The structure is fixed on the inlet end of the channel and free at the outlet end. The velocity inlet flow corresponds to a Reynolds number of 250 in the upper channel and 354 in the lower channel. A pressure outlet with a zero gauge pressure is specified at the outlet, implying that the fluid of the top and bottom channel merge and have the same pressure condition. A fully developed flow is assumed and is specified through the FLUENT user-defined function `fsi_channel_2d.c` for two-dimensional problems and `fsi_channel_3d.c` for three-dimensional problems.

Loading: The fluid flow induces both normal pressure and viscous shear forces on the cantilever. The viscous shear forces are relatively small. The cantilever deforms due to the pressure difference in the top and bottom channels.

Analytical results: A fully developed flow is assumed through the uniform cross-section channel with an incompressible fluid. Thus, the *y*-velocity component (u_y) and the gradient of the *x*-velocity component ($\frac{\partial u_x}{\partial y}$) are zero everywhere; and the governing Navier-Stokes equation for the fluid flow is

$$\frac{\partial p}{\partial x} = \mu \frac{\partial^2 u_x}{\partial y^2},$$

where *y* represents a local coordinate system of each channel, $y = 0$ represents the cantilever interface, and $y = h$ represents the channel wall. The flow at the fluid-structure interface and the channel wall are zero. Thus, $u_x = 0$ at $y = 0$ and $y = h$ for both the top and bottom channels.

Substituting the boundary condition and integrating the Navier-Stokes equation leads to the flow solution for each channel:

$$u(y) = \frac{1}{2\mu}\frac{\partial p}{\partial x}\left(y^2 - hy\right).$$

The mean velocity, u_m, is defined as the integral of the flow solution over the channel cross-sectional area divided by the cross-sectional area. Assuming a unit depth,

$$u_m = \frac{1}{12\mu}\frac{\partial p}{\partial x}h^2.$$

Solving for the pressure gradient, you obtain a linear pressure distribution in each channel,

$$p(x) = 12\frac{\mu u_m}{h^2}(L - x) + p_o,$$

where p_o is the gauge pressure at the outlet.

The deformation of a cantilever beam subjected to a triangular distributed pressure is given by

$$\delta(x) = \frac{P_{max}}{120EIL^2}\left(x^5 - 5Lx^4 + 10L^2x^3 - 10L^3x^2\right),$$

where P_{max} is the pressure at the inlet end.

The tip deflection due to the flow in each channel is

$$\delta_{tip} = -\frac{P_{max}L^4}{30EI}.$$

Since the flow fields merge and the structure is linear, you can superimpose the results for both channels.

Units: The SI unit system is used. Abaqus does not require that the analysis be run with a particular unit system as long as all properties are specified in a consistent manner. However, the unit system used by Abaqus must coincide with those used by the third-party analysis code.

Coupling scheme

A unidirectional coupling scheme, illustrated in *Figure 2*, is employed with FLUENT designated to begin the exchange process by sending its exchange information first. FLUENT computes the flow field around the undeformed cantilever (arrow 1) and sends the pressure distribution to Abaqus (arrow 2). Abaqus then computes the deformation corresponding to the pressure field during the first increment (arrow 3).

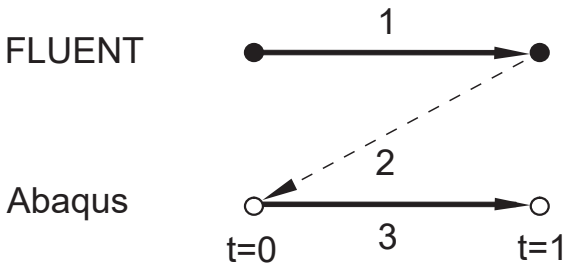


Figure 2: Coupling scheme for unidirectional simulation.

Running the co-simulation

The following procedure illustrates how to run the co-simulation using the MpCCI project file:

- The Abaqus and FLUENT problem files should be copied to the appropriate product directories: *problemDir*/ABAQUS and *problemDir*/FLUENT, and the MpCCI project file should be copied into the *problemDir* directory.
- From the *problemDir* directory, submit the MpCCI project file to MpCCI GUI in batch mode:

```
mpcci -batch example.csp
```

Results and discussion

Based on the analytical derivation for normal pressure distribution, the expected tip deflection is -1.235×10^{-4} m. The simulation results are shown in *Table 1* for a case in which normal pressure (PRESS) is imported into Abaqus and a case in which concentrated forces (CF) are imported into Abaqus.

Table 1: Results for unidirectional transfer.

Element	Tip Deflection (m) (PRESS)	Tip Deflection (m) (CF)
CPS4I	-1.202×10^{-4} (–2.7%)	-1.202×10^{-4} (–2.7%)

The pressure difference between the top and bottom channels reported by FLUENT shows a –2.7% difference compared with the analytically predicted pressure difference. This discrepancy is consistent with the differences observed in the tip deflections. Viscous shear forces, which are not consistent with the analytical derivation, are transferred in addition to the normal pressure forces for cases in which concentrated forces are exchanged. These viscous shear forces are relatively small.

Input files

Unidirectional transfer

fsi_channel_cps4i_pr_1-way.inp	Abaqus input file for unidirectional transfer with pressure loads imported.
fsi_channel_cps4i_cf_1-way.inp	Abaqus input file for unidirectional transfer with concentrated forces imported.
fsi_channel_cps4i_pr_1-way.csp	MpCCI GUI project file for fsi_channel_cps4i_pr_1-way.inp.
fsi_channel_cps4i_cf_1-way.csp	MpCCI GUI project file for fsi_channel_cps4i_cf_1-way.inp.
fsi_channel_2d.cas	FLUENT case file for all two-dimensional models.
fsi_channel_2d_1-way.jou	FLUENT journal file for all unidirectional transfers.
fsi_channel_2d.c	FLUENT user-defined function for two-dimensional laminar flow.

Bidirectional solution transfer between Abaqus/Standard and FLUENT

Elements tested

CPS4I C3D8I

Features tested

- Bidirectional solution transfer between Abaqus/Standard and FLUENT;
- transfer of current coordinates to FLUENT and pressure and concentrated forces to Abaqus;
- two-dimensional and three-dimensional simulations;
- serial and parallel coupling schemes; and
- nodal transformations.

Problem description

Model: The two-dimensional model is identical to the model used for the unidirectional solution transfer. A three-dimensional model is included and described under this section. In addition, two-dimensional and three-dimensional models with nodal transformations specified at the fluid-structure interface are included.

Mesh: The three-dimensional structural mesh consists of continuum elements: 100 elements along the length, and 4 elements stacked in the thickness direction. No mesh parameter studies were performed on the structural mesh.

The fluid mesh for the three-dimensional model consists of 200 hexahedron cells along the channel length and 8 cells and 16 cells stacked in the top channel and bottom channels, respectively. Quadrilateral fluid cells were used since these generally provide better surface pressures than prismatic fluid cells.

Boundary conditions: The boundary conditions are identical to the boundary conditions specified for the unidirectional solution transfer.

Loading: The fluid flow over the channel induces both normal pressure and viscous shear forces on the cantilever. The viscous shear forces are relatively small. The cantilever deforms in response to the pressure differential between the flow in the top and bottom channels. The deformations are transferred back to FLUENT,

and a new flow solution is obtained. This process is repeated until a steady-state condition is established; specifically, until minor changes in deformation and pressure are observed between consecutive coupling steps.

Analytical results: The formulation derived under the unidirectional solution transfer holds only if there is no significant cross-flow; i.e., no flow perpendicular to the cantilever. As the deflection of the cantilever increases, the cross-flow becomes more dominant and, thus, the numerical results deviate from the analytical results.

Coupling schemes

The simulations are run using both serial and parallel coupling schemes illustrated in *Figure 3* and *Figure 4*, respectively.

For the serial coupling scheme FLUENT computes the flow field around the undeformed cantilever (arrow 1). The pressure is transferred to Abaqus (arrow 2). Abaqus computes the deformation corresponding to the pressure field during the first increment and sends the deformed configuration to FLUENT (arrows 3 and 4). This completes one coupling step. FLUENT then computes a new flow solution based on the current configuration of the cantilever (arrow 5), and the steps are repeated until a steady solution is obtained. Typically, only a few exchanges are needed until solutions quantities show minor differences between consecutive coupling steps.

For the parallel coupling scheme FLUENT computes the flow field around the undeformed cantilever (arrow 1) and Abaqus performs an initial increment without any FSI loads. When the target time is reached, both analysis codes exchange solution quantities (arrow 2). Abaqus and FLUENT independently proceed to compute a new solution based on the quantities received from the previous coupling step. Typically, only a few exchanges are needed until the solutions quantities show minor differences between consecutive coupling steps.

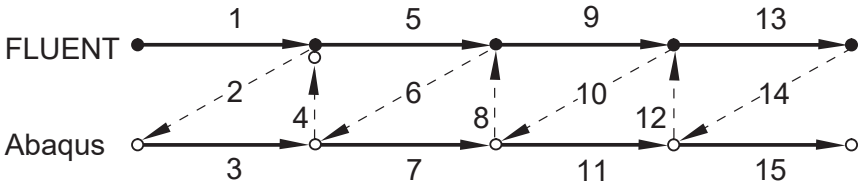


Figure 3: Serial coupling scheme.

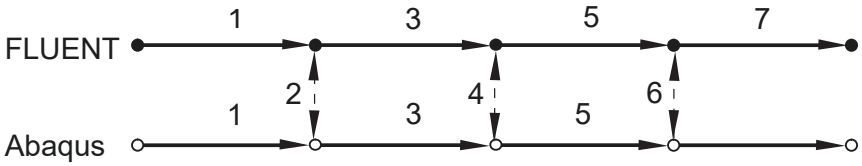


Figure 4: Parallel coupling scheme.

Running the co-simulation

The following procedure illustrates how to run the co-simulation using the MpCCI project file:

- The Abaqus and FLUENT problem files should be copied to the appropriate product directories: *problemDir*/ABAQUS and *problemDir*/FLUENT, and the MpCCI project file should be copied into the *problemDir* directory.
- From the *problemDir* directory, submit the MpCCI project file to MpCCI GUI in batch mode:

  ```
  mpcci -batch example.csp
  ```

The MpCCI configuration files are also included, such that these problems can be run without the MpCCI GUI.

Results and discussion

The solution for the bidirectional transfer is expected to be close to the unidirectional transfer because of the small tip deflection. The simulation results are shown in *Table 2* for the case in which normal pressure (PRESS) is imported into Abaqus and for the case in which concentrated forces (CF) are imported into Abaqus.

Table 2: Results for bidirectional transfer.

Element	Tip Deflection (m) (PRESS)	Tip Deflection (m) (CF)
CPS4I (serial)	-1.148×10^{-4}	-1.148×10^{-4}
CPS4I (parallel)	-1.148×10^{-4}	-1.148×10^{-4}
C3D8I (serial)	-1.158×10^{-4}	-1.158×10^{-4}
C3D8I (parallel)	-1.158×10^{-4}	-1.158×10^{-4}
C3D20R (serial)	-1.162×10^{-4}	-1.162×10^{-4}

The input files used with nodal transformation on the fluid-structure interface yield the same solution as the case without nodal transformation, thus verifying that the concentrated loads are properly transformed to the local coordinate system prior to applying the loads.

Input files

Serial coupling scheme

fsi_channel_cps4i_pr_crd.inp
: Abaqus input file using CPS4I elements; bidirectional transfer with pressure loads imported and current coordinates exported.

fsi_channel_cps4i_cf_crd.inp
: Abaqus input file using CPS4I elements; bidirectional transfer with concentrated forces imported and current coordinates exported.

fsi_channel_c3d8i_pr_crd.inp
: Abaqus input file using C3D8I elements; bidirectional transfer with pressure loads imported and current coordinates exported.

fsi_channel_c3d8i_cf_crd.inp
: Abaqus input file using C3D8I elements; bidirectional transfer with concentrated forces imported and current coordinates exported.

fsi_channel_c3d20r_cf_crd.inp
: Abaqus input file using C3D20R elements; bidirectional transfer with concentrated forces imported and current coordinates exported.

fsi_channel_c3d20r_cf_crd.inp
: Abaqus input file using C3D20R elements; bidirectional transfer with concentrated forces imported and current coordinates exported.

fsi_channel_cps4i_pr_crd.csp
: MpCCI GUI project file for fsi_channel_cps4i_pr_crd.inp.

fsi_channel_cps4i_cf_crd.csp
: MpCCI GUI project file for fsi_channel_cps4i_pr_crd.inp.

fsi_channel_c3d8i_pr_crd.csp
: MpCCI GUI project file for fsi_channel_cps4i_pr_crd_par.inp.

fsi_channel_c3d8i_cf_crd.csp
: MpCCI GUI project file for fsi_channel_c3d8i_cf_crd_par.inp.

fsi_channel_c3d20r_pr_crd.csp
: MpCCI GUI project file for fsi_channel_c3d20r_pr_crd.inp.

fsi_channel_c3d20r_cf_crd.csp
: MpCCI GUI project file for fsi_channel_c3d20r_cf_crd.inp.

fsi_channel_2d_transient.cas
: FLUENT case file for all two-dimensional problems.

fsi_channel_2d.jou
: FLUENT journal file for all two-dimensional problems.

fsi_channel_2d.c
: FLUENT user-defined function for two-dimensional laminar flow.

fsi_channel_3d_transient.cas
: FLUENT case file for all three-dimensional problems.

fsi_channel_3d.jou
: FLUENT journal file for all three-dimensional problems.

fsi_channel_3d.c FLUENT user-defined function for three-dimensional laminar flow.

Parallel coupling scheme

fsi_channel_cps4i_pr_crd_par.inp Abaqus input file using CPS4I elements; bidirectional transfer with pressure loads imported and current coordinates exported.

fsi_channel_cps4i_cf_crd_par.inp Abaqus input file using CPS4I elements; bidirectional transfer with concentrated forces imported and current coordinates exported.

fsi_channel_c3d8i_pr_crd_par.inp Abaqus input file using C3D8I elements; bidirectional transfer with pressure loads imported and current coordinates exported.

fsi_channel_c3d8i_cf_crd_par.inp Abaqus input file using C3D8I elements; bidirectional transfer with concentrated forces imported and current coordinates exported.

fsi_channel_cps4i_pr_crd_par.csp MpCCI GUI project file for fsi_channel_cps4i_pr_crd_par.inp.

fsi_channel_cps4i_cf_crd_par.csp MpCCI GUI project file for fsi_channel_cps4i_cf_crd_par.inp.

fsi_channel_c3d8i_pr_crd_par.csp MpCCI GUI project file for fsi_channel_c3d8i_pr_coord_par.inp.

fsi_channel_c3d8i_cf_crd_par.csp MpCCI GUI project file for fsi_channel_c3d8i_cf_crd_par.inp.

fsi_channel_2d_transient.cas FLUENT case file for all two-dimensional problems.

fsi_channel_2d_par.jou FLUENT journal file for all two-dimensional problems.

fsi_channel_2d.c FLUENT user-defined function for two-dimensional laminar flow.

fsi_channel_3d_transient.cas FLUENT case file for all three-dimensional problems.

fsi_channel_3d_par.jou FLUENT journal file for all three-dimensional problems.

fsi_channel_3d.c FLUENT user-defined function for three-dimensional laminar flow.

Nodal transformation

fsi_channel_cps4i_cf_crd_trnsf.inp Abaqus input file using CPS4I elements; bidirectional transfer with concentrated forces imported and current coordinates exported using nodal transformation.

fsi_channel_c3d8i_cf_crd_trnsf.inp Abaqus input file using C3D8I elements; bidirectional transfer with concentrated forces imported and current coordinates exported using nodal transformation.

fsi_channel_cps4i_cf_crd_trnsf.csp MpCCI GUI project file for fsi_channel_cps4i_cf_crd_trnsf.inp.

fsi_channel_c3d8i_cf_crd_trnsf.csp MpCCI GUI project file for fsi_channel_c3d8i_cf_crd_trnsf.inp.

fsi_channel_2d_transient.cas FLUENT case file for all two-dimensional problems.

fsi_channel_2d.jou FLUENT journal file for all two-dimensional problems.

fsi_channel_2d.c FLUENT user-defined function for two-dimensional laminar flow.

fsi_channel_3d_transient.cas FLUENT case file for all three-dimensional problems.

fsi_channel_3d.jou FLUENT journal file for all three-dimensional problems.

fsi_channel_3d.c FLUENT user-defined function for three-dimensional laminar flow.

Rendezvousing scheme

Elements tested

CPS4I

Features tested

The following rendezvousing schemes are tested in Abaqus/Standard:

- The coupling step size is a user-defined constant and Abaqus/Standard is forced to use a single increment per coupling step (lockstep).
- The coupling step size is a user-defined constant and Abaqus/Standard is allowed to take one or more increments during the coupling step (subcycle).
- The coupling step size is defined by FLUENT and Abaqus/Standard is allowed to take one or more increments during the coupling step (subcycle).

Problem description

The problem is identical to the two-dimensional channel problem discussed in the previous sections, with the exception of the time stepping scheme. The rendezvousing scheme is defined through the MpCCI GUI. Specifying a target time period allows Abaqus to subcycle based on its own time stepping scheme while maintaining exchanges with the third-party code at a fixed frequency. Abaqus/Standard interpolates the imported loads between the previous coupling step and the target values.

Running the co-simulation

The following procedure illustrates how to run the co-simulation using the MpCCI project file:

- The Abaqus and FLUENT problem files should be copied to the appropriate product directories: *problemDir*/ABAQUS and *problemDir*/FLUENT, and the MpCCI project file should be copied into the *problemDir* directory.
- From the *problemDir* directory, submit the MpCCI project file to MpCCI GUI in batch mode:

```
mpcci -batch example.csp
```

The MpCCI configuration files are also included, such that these problems can be run from without the MpCCI GUI.

Results and discussion

The loads are properly interpolated during subcycles, and the rendezvous times are met as specified by the rendezvousing scheme. This has been verified by plotting a history plot of the variable CF at an interface node.

Input files

fsi_channel_cps4i_constantDt_lockstep.inp	Abaqus input file where the coupling step size is a user-defined constant and Abaqus/Standard is forced to use a single increment per coupling step (lockstep).
fsi_channel_cps4i_constantDt.inp	Abaqus input file using C3D8I elements; bidirectional transfer, automatic time stepping, and meeting target times in a loose manner.
fsi_channel_cps4i_importDt.inp	Abaqus input file using C3D8I elements; bidirectional transfer, direct user-specified time stepping, and meeting target times exactly.
fsi_channel_cps4i_constantDt_lockstep.csp	MpCCI GUI project file for fsi_channel_cps4i_constantDt_lockstep.inp.
fsi_channel_cps4i_constantDt.csp	MpCCI GUI project file for fsi_channel_cps4i_constantDt.inp.
fsi_channel_cps4i_importDt.csp	MpCCI GUI project file for fsi_channel_cps4i_importDt.inp.

fsi_channel_2d_transient.cas	FLUENT case file for all two-dimensional problems.
fsi_channel_2d.jou	FLUENT journal file for all two-dimensional problems.
fsi_channel_2d.c	FLUENT user-defined function for two-dimensional laminar flow.

Abaqus/Standard to Abaqus/Explicit co-simulation

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

The tests in this section verify the co-simulation interaction of Abaqus/Standard and Abaqus/Explicit when the two analysis products address complementary domains of a model. Results obtained from an Abaqus/Explicit simulation of the same model are used as a reference solution.

Features tested

The following sections describe verification problems for:

- lockstep co-simulation of Abaqus/Explicit with Abaqus/Standard nonlinear dynamic procedures;
- subcycling co-simulation of Abaqus/Explicit with Abaqus/Standard nonlinear dynamic procedures;
- subcycling co-simulation of Abaqus/Explicit with Abaqus/Standard nonlinear quasi-static procedures; and
- various cases of modeling techniques and model attributes applied at the co-simulation interface between the Abaqus/Explicit and Abaqus/Standard jobs.

Lockstep co-simulation of Abaqus/Standard nonlinear dynamic procedures to Abaqus/Explicit procedures

Elements tested

B31 C3D8I C3D8 C3D4 S4R T3D2

Features tested

The fidelity and numerical stability of results obtained using a lockstep Abaqus/Standard to Abaqus/Explicit co-simulation for a model undergoing dynamic large-deformation motion.

Problem description

The problem is a simple beam subjected to an excitation force at the end (see *Figure 1*).

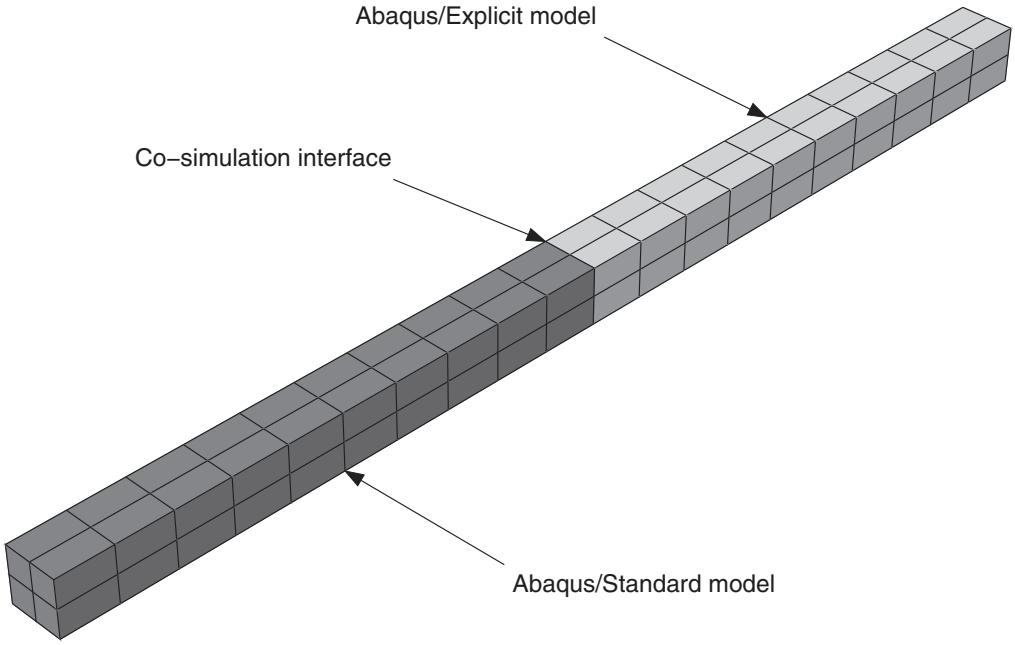


Figure 1: Continuum element co-simulation model configuration.The configuration of the beam verification problems lies on the centerline of the models shown in this figure. Shell elements, when used, lie on the outside of the models shown.

Model: The model consists of Abaqus/Standard and Abaqus/Explicit components of a beam of length 20, width 1, and height 1.

Mesh: A regular brick mesh is used for the continuum and shell element models.

Material: A linear elastic material definition is used.

Boundary conditions: The Abaqus/Standard portion of the beam is fully embedded at its end.

Loading: The Abaqus/Explicit portion of the beam has a load applied transverse to the beam axis.

Co-simulation definition: Each model uses the lockstep method on the co-simulation controls.

Lockstep co-simulation algorithm description

When using the lockstep method, Abaqus/Standard and Abaqus/Explicit will advance their respective solution using the same time incrementation.

Results and discussion

In each case there is generally good agreement between the Abaqus/Standard to Abaqus/Explicit co-simulation results and the Abaqus/Explicit results.

Input files

Beam element tests:

beam_dyntodyn_lockstep_std.inp	B31Abaqus/Standard analysis.
beam_dyntodyn_lockstep_xpl.inp	B31Abaqus/Explicit analysis.
beam_dyntodyn_lockstep_config.xml	Co-simulation configuration file.
beam_fullxpl.inp	B31Abaqus/Explicit reference analysis.

Continuum element tests:

contbeam_dyntodyn_lockstep_std.inp	C3D8IAbaqus/Standard analysis.
contbeam_dyntodyn_lockstep_xpl.inp	C3D8IAbaqus/Explicit analysis.
contbeam_dyntodyn_lockstep_config.xml	Co-simulation configuration file.
contbeam_fullxpl.inp	C3D8IAbaqus/Explicit reference analysis.

Mixed element tests:

contbeam_rot_dyntodyn_lockstep_std.inp	B31, C3D8I, S4RAbaqus/Standard analysis.
contbeam_rot_dyntodyn_lockstep_xpl.inp	B31, C3D8I, S4RAbaqus/Explicit analysis.
contbeam_rot_dyntodyn_lockstep_config.xml	Co-simulation configuration file.
contbeam_rot_fullxpl.inp	B31, C3D8I, S4RAbaqus/Explicit reference analysis.

Truss element tests with axial loading:

truss_dyntodyn_lockstep_std.inp	T3D2Abaqus/Standard analysis.
truss_dyntodyn_lockstep_xpl.inp	T3D2Abaqus/Explicit analysis.
truss_dyntodyn_lockstep_config.xml	Co-simulation configuration file.
truss_fullxpl.inp	T3D2Abaqus/Explicit reference analysis.

Tests with dissimilar meshes at the co-simulation interface regions:

contbeam_dmesh_dyntodyn_lock_std.inp	C3D4Abaqus/Standard analysis.
contbeam_dmesh_dyntodyn_lock_xpl.inp	C3D8Abaqus/Explicit analysis.
contbeam_dmesh_dyntodyn_lock_config.xml	Co-simulation configuration file.
contbeam_dmesh2_dyntodyn_lock_std.inp	C3D8Abaqus/Standard analysis.
contbeam_dmesh2_dyntodyn_lock_xpl.inp	C3D4Abaqus/Explicit analysis.
contbeam_dmesh2_dyntodyn_lock_config.xml	Co-simulation configuration file.

Subcycling co-simulation of Abaqus/Standard nonlinear dynamic procedures to Abaqus/Explicit procedures

Elements tested

B31 C3D8I C3D8 C3D4 S4R T3D2

Features tested

The fidelity and numerical stability of results obtained using a subcycling Abaqus/Standard to Abaqus/Explicit co-simulation for a model undergoing dynamic large-deformation motion.

Problem description

The problem is a simple beam subjected to severe excitation force (see *Figure 1*).

Model: The model consists of Abaqus/Standard and Abaqus/Explicit components of a beam of length 20, width 1, and height 1.

Mesh: A regular brick mesh is used for the continuum and shell element models.

Material: A linear elastic material definition is used.

Boundary conditions: The Abaqus/Standard portion of the beam is fully embedded at its end.

Loading: The Abaqus/Explicit portion of the beam has a load applied transverse to the beam axis.

Co-simulation definition: Each model uses the subcycle method on the co-simulation controls.

Subcycling co-simulation algorithm description

When using the subcycle method, Abaqus/Standard and Abaqus/Explicit will advance their respective solutions using time incrementation appropriate to their solution.

Results and discussion

In each case there is generally good agreement between the Abaqus/Standard to Abaqus/Explicit co-simulation results and the Abaqus/Explicit results.

Input files

Beam element tests:

beam_dyntodyn_subcycle_std.inp	B31Abaqus/Standard analysis.
beam_dyntodyn_subcycle_xpl.inp	B31Abaqus/Explicit analysis.
beam_dyntodyn_subcycle_config.xml	Co-simulation configuration file.
beam_fullxpl.inp	B31Abaqus/Explicit reference analysis.

Continuum element tests:

contbeam_dyntodyn_subcycle_std.inp	C3D8IAbaqus/Standard analysis.
contbeam_dyntodyn_subcycle_xpl.inp	C3D8IAbaqus/Explicit analysis.
contbeam_dyntodyn_subcycle_config.xml	Co-simulation configuration file.
contbeam_fullxpl.inp	C3D8IAbaqus/Explicit reference analysis.

Mixed element tests:

contbeam_rot_dyntodyn_subcycle_std.inp	B31, C3D8I, S4RAbaqus/Standard analysis.
contbeam_rot_dyntodyn_subcycle_xpl.inp	B31, C3D8I, S4RAbaqus/Explicit analysis.
contbeam_rot_dyntodyn_subcycle_config.xml	Co-simulation configuration file.
contbeam_rot_fullxpl.inp	B31, C3D8I, S4RAbaqus/Explicit reference analysis.

Truss element tests with axial loading:

truss_dyntodyn_subcycle_std.inp	T3D2Abaqus/Standard analysis.
truss_dyntodyn_subcycle_xpl.inp	T3D2Abaqus/Explicit analysis.
truss_dyntodyn_subcycle_config.xml	Co-simulation configuration file.
truss_fullxpl.inp	T3D2Abaqus/Explicit reference analysis.

Tests with dissimilar meshes at the co-simulation interface regions:

contbeam_dmesh_dyntodyn_sub_std.inp	C3D4Abaqus/Standard analysis.
contbeam_dmesh_dyntodyn_sub_xpl.inp	C3D8Abaqus/Explicit analysis.

contbeam_dmesh_dyntodyn_sub_config.xml	Co-simulation configuration file.
contbeam_dmesh2_dyntodyn_sub_std.inp	C3D8Abaqus/Standard analysis.
contbeam_dmesh2_dyntodyn_sub_xpl.inp	C3D4Abaqus/Explicit analysis.
contbeam_dmesh2_dyntodyn_sub_config.xml	Co-simulation configuration file.

Tests of less frequent interface matrix factorization:The following input files test the factorization of the interface matrix once per Abaqus/Standard increment on the co-simulation controls.

contbeam_dyntodyn_fact_std.inp	C3D8IAbaqus/Standard analysis.
contbeam_dyntodyn_fact_xpl.inp	C3D8IAbaqus/Explicit analysis.
contbeam_dyntodyn_fact_config.xml	Co-simulation configuration file.
contbeam_dmesh_dyntodyn_fact_std.inp	C3D4Abaqus/Standard analysis.
contbeam_dmesh_dyntodyn_fact_xpl.inp	C3D8Abaqus/Explicit analysis.
contbeam_dmesh_dyntodyn_fact_config.xml	Co-simulation configuration file.

Subcycling co-simulation of Abaqus/Standard nonlinear static procedures to Abaqus/Explicit procedures

Elements tested

B31 C3D8I C3D8 C3D4 S4R T3D2

Features tested

The fidelity and numerical stability of results obtained using subcycling Abaqus/Standard quasi-static procedures to Abaqus/Explicit co-simulation for a model undergoing quasi-static deformation.

Problem description

The problem is a simple beam subjected to quasi-static loading (see *Figure 1*).

Model: The model consists of Abaqus/Standard and Abaqus/Explicit components of a beam of length 20, width 1, and height 1.

Mesh: A regular brick mesh is used for the continuum and shell element models.

Material: A linear elastic material definition is used.

Boundary conditions: The Abaqus/Standard portion of the beam is fully embedded at the free end.

Loading: The Abaqus/Explicit portion of the beam has a load applied transverse to the beam axis.

Co-simulation definition: Each model uses the subcycle method on the co-simulation controls.

Subcycling co-simulation algorithm description

When using the subcycle method, Abaqus/Standard and Abaqus/Explicit will advance their respective solutions using time incrementation appropriate to their solution.

Results and discussion

In each case there is generally good agreement between the Abaqus/Standard to Abaqus/Explicit co-simulation results and the Abaqus/Explicit results.

Input files

Continuum element tests:

contbeam_statodyn_subcycle_std.inp	C3D8IAbaqus/Standard analysis.
contbeam_statodyn_subcycle_xpl.inp	C3D8IAbaqus/Explicit analysis.
contbeam_statodyn_subcycle_config.xml	Co-simulation configuration file.
contbeam_quasistatic_fullxpl.inp	C3D8IAbaqus/Explicit reference analysis.

Mixed element tests:

contbeam_rot_statodyn_subcycle_std.inp	B31, C3D8I, S4RAbaqus/Standard analysis.
contbeam_rot_statodyn_subcycle_xpl.inp	B31, C3D8I, S4RAbaqus/Explicit analysis.
contbeam_rot_statodyn_fact_config.xml	Co-simulation configuration file.
contbeam_rot_quasistatic_fullxpl.inp	B31, C3D8I, S4RAbaqus/Explicit reference analysis.

Truss element tests with axial loading:

truss_statodyn_subcycle_std.inp	T3D2Abaqus/Standard analysis.
truss_statodyn_subcycle_xpl.inp	T3D2Abaqus/Explicit analysis.
truss_statodyn_subcycle_config.xml	Co-simulation configuration file.
truss_fullxpl.inp	T3D2Abaqus/Explicit reference analysis.

Tests with dissimilar meshes at the co-simulation interface regions:

contbeam_dmesh_statodyn_sub_std.inp	C3D4Abaqus/Standard analysis.
contbeam_dmesh_statodyn_sub_xpl.inp	C3D8Abaqus/Explicit analysis.
contbeam_dmesh_statodyn_sub_config.xml	Co-simulation configuration file.
contbeam_dmesh2_statodyn_sub_std.inp	C3D8Abaqus/Standard analysis.
contbeam_dmesh2_statodyn_sub_xpl.inp	C3D4Abaqus/Explicit analysis.
contbeam_dmesh2_statodyn_sub_config.xml	Co-simulation configuration file.

Tests of less frequent interface matrix factorization:The following input files test the factorization of the interface matrix once per Abaqus/Standard increment on the co-simulation controls.

contbeam_rot_statodyn_fact_std.inp	B31, C3D8I, S4RAbaqus/Standard analysis.
contbeam_rot_statodyn_fact_xpl.inp	B31, C3D8I, S4RAbaqus/Explicit analysis.
contbeam_rot_statodyn_fact_config.xml	Co-simulation configuration file.
contbeam_dmesh_statodyn_fact_std.inp	C3D4Abaqus/Standard analysis.
contbeam_dmesh_statodyn_fact_xpl.inp	C3D8Abaqus/Explicit analysis.
contbeam_dmesh_statodyn_fact_config.xml	Co-simulation configuration file.

Model attribute tests for Abaqus/Standard to Abaqus/Explicit co-simulation

Elements tested

B21 C3D8I C3D4 SFM3D4R

Features tested

The proper operation of Abaqus/Standard to Abaqus/Explicit co-simulation is confirmed for cases involving specific modeling attributes.

Problem description

Each problem considered is a variation of those described in *Lockstep co-simulation of Abaqus/Standard nonlinear dynamic procedures to Abaqus/Explicit procedures*. Particular variations are listed in the input file description.

Results and discussion

In each case the results confirm that Abaqus/Standard to Abaqus/Explicit co-simulation operates correctly with particular modeling attributes employed.

Input files

Two-dimensional beam element tests:

beam_2d_dyntodyn_subcycle_std.inp	B21Abaqus/Standard analysis.
beam_2d_dyntodyn_subcycle_xpl.inp	B21Abaqus/Explicit analysis.
beam_2d_dyntodyn_subcycle_config.xml	Co-simulation configuration file.

Abaqus/Standard substructure retained nodes at the interface:

contbeam_substruc_subcycle_std.inp	C3D8IAbaqus/Standard analysis.
contbeam_substructure_gen.inp	C3D8IAbaqus/Standard substructure generation.
contbeam_substruc_subcycle_std.inp	C3D8IAbaqus/Explicit analysis.
contbeam_substruc_subcycle_config.xml	Co-simulation configuration file.

Test with mass scaling in Abaqus/Explicit:

contbeam_dyntodyn_mass_scale_std.inp	C3D8IAbaqus/Standard analysis.
contbeam_dyntodyn_mass_scale_xpl.inp	C3D8IAbaqus/Explicit analysis with mass scaling.
contbeam_dyntodyn_mass_scale_config.xml	Co-simulation configuration file.

Tests with tie constraints at the co-simulation interface:

tie_dyntodyn_lockstep_std.inp	C3D4, SFM3D4RAbaqus/Standard analysis.
tie_dyntodyn_lockstep_xpl.inp	C3D4, SFM3D4RAbaqus/Explicit analysis.
tie_dyntodyn_lockstep_config.xml	Co-simulation configuration file.
tie_dyntodyn_subcycle_std.inp	C3D4, SFM3D4RAbaqus/Standard analysis.
tie_dyntodyn_subcycle_xpl.inp	C3D4, SFM3D4RAbaqus/Explicit analysis.
tie_dyntodyn_subcycle_config.xml	Co-simulation configuration file.

Adaptive remeshing

In this section:

- *Pressurized thick-walled cylinder*
- *Error indicators*

Pressurized thick-walled cylinder

Products: Abaqus/Standard Abaqus/CAE

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D10M CPE3 CPE4R CPE6 CPE8

Features tested

- Iterative mesh optimization using Abaqus/CAE and Abaqus/Standard.
- Error indicator variables in Abaqus/Standard.

Problem description

This verification problem considers the case of pressure applied to a thick-walled linear elastic cylinder. The problem, which has a simple closed-form solution, is used to verify the iterative mesh optimization procedure.

Model: All tests consider a quarter-symmetry model of an infinite extent cylinder with an internal radius of 5 and an external radius of 20. Appropriate symmetry boundary conditions are imposed on the horizontal and vertical surfaces (see *Figure 1*).

Mesh: Adaptivity is used to achieve a final mesh that attempts to reach a target error uniformly. The initial mesh is created with various Abaqus/CAE meshing techniques based on uniform seeding.

Material: The stress distribution in the cylinder is independent of choice of linear elastic material properties; hence, a simple modulus of 1000 and a Poisson's ratio of 0.3 are used.

Boundary conditions: Symmetry boundary conditions are applied.

Loading: A unit pressure is applied to the cylinder interior.

Error indicators: The following error indicator variables are tested:

- ENDENERI
- MISESERI

Sizing methods: The following sizing methods are tested:

- Uniform method
- Minimum/maximum method

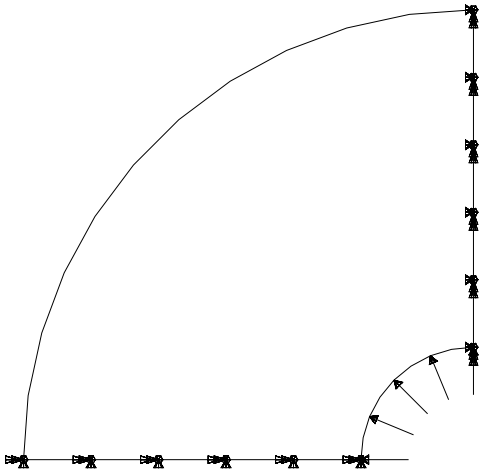


Figure 1: Thick cylinder model.

Results and discussion

The radial and circumferential stress, as well as their radial gradients, vary through the thickness of the cylinder, resulting in a finite element error in stress that varies radially for a uniform initial mesh. Hence, we expect that an optimized mesh, one that results in a radially uniform error, will have a radially varying mesh density.

For the geometry and loading the exact solution for this problem is

$$\sigma_{rr} = -\frac{80}{3r^2} + \frac{1}{15}$$

$$\sigma_{\theta\theta} = \frac{80}{3r^2} + \frac{1}{15}$$

Results are shown in this section for a sequence of plane strain quadrilateral meshes adaptively meshed according to an ENDENERI error indicator variable and the minimum/maximum method sizing approach. Many more element, meshing, and sizing methods are tested in this section; most results, however, are similar to this representative case.

Adaptive remeshing

You can see the progression of meshes in *Figure 2*. Since the gradient in stresses, and consequently the solution error, is higher toward the inside radius, the mesh refinement focuses on the inside radius.

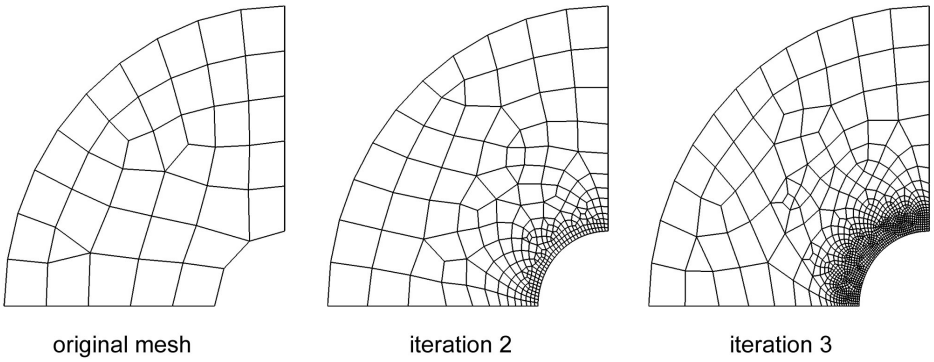


Figure 2: Mesh refinement progression.

Error measures

For each verification problem and mesh iteration the following are calculated:

- The element count for the iteration.
- The computed error indicator.
- The true solution error in σ_{rr}, computed as both a global norm and a peak error.

As you can see from the representative case in *Table 1*, the measure of true error tends to converge more rapidly than the error indicator value.

Table 1: Error measures and indicators.

Iteration	Element count	Error indicator	Error measure	Peak error
1	38	0.479	0.095	0.055
2	272	0.174	0.016	0.033
3	1014	0.146	0.012	0.016

Files

Input files are in the form of Python scripts that you can run in Abaqus/CAE and a user subroutine file that computes the true error at each material point. The scripts will create the model and run an adaptivity analysis sequence of jobs. The input files are named according to a convention that reflects various parameter settings.

Plane strain elements

adaptcyl_cpe4r_E_GL240.py Quadrilateral dominant mesh with CPE4R and CPE3 elements. ENDENERI error indicator. Mimimum/maximum sizing method with 40% target on low-stress errors and a 2% target on high-stress errors.

adaptcyl_cpe8_M_E2.py Quadrilateral dominant mesh with CPE8 and CPE6 elements. MISESERI error indicator. Mimimum/maximum sizing method with 2% target on low-stress errors and a 0.1% target on high-stress errors.

Three-dimensional elements

adaptcyl_c3d10m_E_U5.py Tetrahedral mesh with C3D10M elements. ENDENERI error indicator. Uniform sizing method with 5% target error.

User subroutine file

adaptivity-cylinder.f User subroutines *UVARM* and *UEXTERNALDB*, which calculate the actual error at each material point and report global norms of error.

References

- Timoshenko, S. P., and J. N. Goodier, Theory of Elasticity, McGraw-Hill, 1951.

Error indicators

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

CPS3 CPS3E CPS4 CPS4E CPS4I CPS4R CPS4RT CPS4T CPS6 CPS6E CPS6M
CPS6MT CPS8 CPS8R CPS8RE CPS8RT CPS8T
CPE3 CPE3E CPE3H CPE4 CPE4E CPE4H CPE4HT CPE4I CPE4IH CPE4R
CPE4RH CPE4RHT CPE4RT CPE4T CPE6 CPE6E CPE6H CPE6M CPE6MH
CPE6MHT CPE6MT CPE8 CPE8E CPE8H CPE8HT CPE8R CPE8RE
CPE8RH CPE8RHT CPE8RT CPE8T
CPEG3 CPEG3H CPEG3T CPEG4 CPEG4H CPEG4HT
CPEG4I CPEG4IH CPEG4R
CPEG4RH CPEG4RHT
CPEG4T CPEG6 CPEG6H CPEG6M CPEG6MH CPEG6MHT CPEG6MT CPEG8
CPEG8H CPEG8HT CPEG8R CPEG8RH CPEG8RHT CPEG8T
CAX3 CAX3H CAX4 CAX4H CAX4HT CAX4I CAX4IH CAX4R
CAX4RH CAX4RT
CAX4T CAX6 CAX6H CAX6M CAX6MH CAX6MHT CAX8 CAX8H CAX8HT
CAX8R CAX8RH CAX8RHT CAX8RT CAX8T CGAX4 CGAX6M
C3D4 C3D4E C3D4H C3D10 C3D10E C3D10H C3D10M C3D10MH
C3D10MHT C3D10MT C3D8 C3D8R C3D20 C3D20R
DC2D3 DC2D3E DC2D4 DC2D4E DC2D6 DC2D6E DC2D8 DC2D8E
DC3D10 DC3D10E DC3D4 DC3D4E
S3 S3R S4 S4R S4R5 S8R S8R5 S8RT STRI3 STRI65

Features tested

The following error indicators and element average output are tested.

ENDEN Element energy density.

ENDENERI Element energy density error indicator.

MISESAVG Element average Mises equivalent stress.

MISESERI Mises equivalent stress error indicator.

PEEQAVG Element average equivalent plastic strain.

PEEQERI Equivalent plastic strain error indicator.

PEAVG Element average plastic strain.

PEERI	Plastic strain error indicator.
CEAVG	Element average creep strain.
CEERI	Creep strain error indicator.
HFLAVG	Element average heat flux.
HFLERI	Heat flux error indicator.
EFLAVG	Element average electric flux.
EFLERI	Electric flux error indicator.
EPGAVG	Element average electric potential gradient.
EPGERI	Electric potential gradient error indicator.

Problem description

All the problems have small coarse meshes with solution gradient risers. Sharp solution gradients are provided by stress concentrations, concentrated heat flux, localized plasticity, etc. Various material types are used to test all the supported element types, and the error indicator appropriate for the material properties is output.

Results and discussion

Error indicators have the highest value where the solution gradients are highest, which is confirmed in the verification tests.

Input files

err_cps3.inp	Tests error indicator output for CPS3 elements.
err_cps3e.inp	Tests error indicator output for CPS3E elements.
err_cps4.inp	Tests error indicator output for CPS4 elements.
err_cps4e.inp	Tests error indicator output for CPS4E elements.
err_cps4i.inp	Tests error indicator output for CPS4I elements.
err_cps4r.inp	Tests error indicator output for CPS4R elements.
err_cps4rt.inp	Tests error indicator output for CPS4RT elements.
err_cps4t.inp	Tests error indicator output for CPS4T elements.
err_cps6.inp	Tests error indicator output for CPS6 elements.
err_cps6e.inp	Tests error indicator output for CPS6E elements.
err_cps6m.inp	Tests error indicator output for CPS6M elements.
err_cps6mt.inp	Tests error indicator output for CPS6MT elements.
err_cps8.inp	Tests error indicator output for CPS8 elements.
err_cps8r.inp	Tests error indicator output for CPS8R elements.
err_cps8re.inp	Tests error indicator output for CPS8RE elements.

err_cps8rt.inp	Tests error indicator output for CPS8RT elements.
err_cps8t.inp	Tests error indicator output for CPS8T elements.
err_cpe3.inp	Tests error indicator output for CPE3 elements.
err_cpe3e.inp	Tests error indicator output for CPE3E elements.
err_cpe3h.inp	Tests error indicator output for CPE3H elements.
err_cpe4.inp	Tests error indicator output for CPE4 elements.
err_cpe4e.inp	Tests error indicator output for CPE4E elements.
err_cpe4h.inp	Tests error indicator output for CPE4H elements.
err_cpe4ht.inp	Tests error indicator output for CPE4HT elements.
err_cpe4i.inp	Tests error indicator output for CPE4I elements.
err_cpe4ih.inp	Tests error indicator output for CPE4IH elements.
err_cpe4r.inp	Tests error indicator output for CPE4R elements.
err_cpe4rh.inp	Tests error indicator output for CPE4RH elements.
err_cpe4rht.inp	Tests error indicator output for CPE4RHT elements.
err_cpe4rt.inp	Tests error indicator output for CPE4RT elements.
err_cpe4t.inp	Tests error indicator output for CPE4T elements.
err_cpe6.inp	Tests error indicator output for CPE6 elements.
err_cpe6h.inp	Tests error indicator output for CPE6H elements.
err_cpe6m.inp	Tests error indicator output for CPE6M elements.
err_cpe6mh.inp	Tests error indicator output for CPE6MH elements.
err_cpe6mht.inp	Tests error indicator output for CPE6MHT elements.
err_cpe6mt.inp	Tests error indicator output for CPE6MT elements.
err_cpe8.inp	Tests error indicator output for CPE8 elements.
err_cpe8e.inp	Tests error indicator output for CPE8E elements.
err_cpe8h.inp	Tests error indicator output for CPE8H elements.
err_cpe8ht.inp	Tests error indicator output for CPE8HT elements.
err_cpe8r.inp	Tests error indicator output for CPE8R elements.
err_cpe8re.inp	Tests error indicator output for CPE8RE elements.
err_cpe8rh.inp	Tests error indicator output for CPE8RH elements.
err_cpe8rht.inp	Tests error indicator output for CPE8RHT elements.
err_cpe8t.inp	Tests error indicator output for CPE8T elements.
err_cpeg3.inp	Tests error indicator output for CPEG3 elements.
err_cpeg3h.inp	Tests error indicator output for CPEG3H elements.
err_cpeg3ht.inp	Tests error indicator output for CPEG3HT elements.
err_cpeg3t.inp	Tests error indicator output for CPEG3T elements.
err_cpeg4.inp	Tests error indicator output for CPEG4 elements.
err_cpeg4h.inp	Tests error indicator output for CPEG4H elements.
err_cpeg4ht.inp	Tests error indicator output for CPEG4HT elements.
err_cpeg4i.inp	Tests error indicator output for CPEG4I elements.
err_cpeg4ih.inp	Tests error indicator output for CPEG4IH elements.
err_cpeg4r.inp	Tests error indicator output for CPEG4R elements.
err_cpeg4rh.inp	Tests error indicator output for CPEG4RH elements.
err_cpeg4rht.inp	Tests error indicator output for CPEG4RHT elements.

err_cpeg4t.inp	Tests error indicator output for CPEG4T elements.
err_cpeg6.inp	Tests error indicator output for CPEG6 elements.
err_cpeg6h.inp	Tests error indicator output for CPEG6H elements.
err_cpeg6m.inp	Tests error indicator output for CPEG6M elements.
err_cpeg6mh.inp	Tests error indicator output for CPEG6MH elements.
err_cpeg6mht.inp	Tests error indicator output for CPEG6MHT elements.
err_cpeg6mt.inp	Tests error indicator output for CPEG6MT elements.
err_cpeg8.inp	Tests error indicator output for CPEG8 elements.
err_cpeg8h.inp	Tests error indicator output for CPEG8H elements.
err_cpeg8ht.inp	Tests error indicator output for CPEG8HT elements.
err_cpeg8r.inp	Tests error indicator output for CPEG8R elements.
err_cpeg8rh.inp	Tests error indicator output for CPEG8RH elements.
err_cpeg8rht.inp	Tests error indicator output for CPEG8RHT elements.
err_cpeg8t.inp	Tests error indicator output for CPEG8T elements.
err_cax3.inp	Tests error indicator output for CAX3 elements.
err_cax3h.inp	Tests error indicator output for CAX3H elements.
err_cax4.inp	Tests error indicator output for CAX4 elements.
err_cax4h.inp	Tests error indicator output for CAX4H elements.
err_cax4ht.inp	Tests error indicator output for CAX4HT elements.
err_cax4i.inp	Tests error indicator output for CAX4I elements.
err_cax4ih.inp	Tests error indicator output for CAX4IH elements.
err_cax4r.inp	Tests error indicator output for CAX4R elements.
err_cax4rh.inp	Tests error indicator output for CAX4RH elements.
err_cax4rt.inp	Tests error indicator output for CAX4RT elements.
err_cax6.inp	Tests error indicator output for CAX6 elements.
err_cax6h.inp	Tests error indicator output for CAX6H elements.
err_cax6m.inp	Tests error indicator output for CAX6M elements.
err_cax6mh.inp	Tests error indicator output for CAX6MH elements.
err_cax6mht.inp	Tests error indicator output for CAX6MHT elements.
err_cax8.inp	Tests error indicator output for CAX8 elements.
err_cax8h.inp	Tests error indicator output for CAX8H elements.
err_cax8ht.inp	Tests error indicator output for CAX8HT elements.
err_cax8r.inp	Tests error indicator output for CAX8R elements.
err_cax8rh.inp	Tests error indicator output for CAX8RH elements.
err_cax8rht.inp	Tests error indicator output for CAX8RHT elements.
err_cax8rt.inp	Tests error indicator output for CAX8RT elements.
err_cax8t.inp	Tests error indicator output for CAX8T elements.
err_cgax4.inp	Tests error indicator output for CGAX4 elements.
err_cgax6m.inp	Tests error indicator output for CGAX6M elements.
err_dc2d3.inp	Tests error indicator output for DC2D3 elements.
err_dc2d3e.inp	Tests error indicator output for DC2D3E elements.
err_dc2d4.inp	Tests error indicator output for DC2D4 elements.
err_dc2d4e.inp	Tests error indicator output for DC2D4E elements.

err_dc2d6.inp	Tests error indicator output for DC2D6 elements.
err_dc2d6e.inp	Tests error indicator output for DC2D6E elements.
err_dc2d8.inp	Tests error indicator output for DC2D8 elements.
err_dc2d8e.inp	Tests error indicator output for DC2D8E elements.
err_dc3d10.inp	Tests error indicator output for DC3D10 elements.
err_dc3d10e.inp	Tests error indicator output for DC3D10E elements.
err_dc3d4.inp	Tests error indicator output for DC3D4 elements.
err_dc3d4e.inp	Tests error indicator output for DC3D4E elements.
err_c3d4.inp	Tests error indicator output for C3D4 elements.
err_c3d4e.inp	Tests error indicator output for C3D4E elements.
err_c3d4h.inp	Tests error indicator output for C3D4H elements.
err_c3d10.inp	Tests error indicator output for C3D10 elements.
err_c3d10e.inp	Tests error indicator output for C3D10E elements.
err_c3d10h.inp	Tests error indicator output for C3D10H elements.
err_c3d10m.inp	Tests error indicator output for C3D10M elements.
err_c3d10mh.inp	Tests error indicator output for C3D10MH elements.
err_c3d10mht.inp	Tests error indicator output for C3D10MHT elements.
err_c3d10mt.inp	Tests error indicator output for C3D10MT elements.
err_s3.inp	Tests error indicator output for S3 elements.
err_s3r.inp	Tests error indicator output for S3R elements.
err_s4.inp	Tests error indicator output for S4 elements.
err_s4r.inp	Tests error indicator output for S4R elements.
err_s4r5.inp	Tests error indicator output for S4R5 elements.
err_s8r.inp	Tests error indicator output for S8R elements.
err_s8r5.inp	Tests error indicator output for S8R5 elements.
err_s8rt.inp	Tests error indicator output for S8RT elements.
err_stri3.inp	Tests error indicator output for STRI3 elements.
err_stri65.inp	Tests error indicator output for STRI65 elements.
err_perturb.inp	Tests error indicator output for perturbation steps.
err_c3d8.inp	Tests error indicator output for C3D8 elements.
err_c3d8r.inp	Tests error indicator output for C3D8R elements.
err_c3d20.inp	Tests error indicator output for C3D20 elements.
err_c3d20r.inp	Tests error indicator output for C3D20R elements.
err_parallel.inp	Tests error indicator output for element loop parallel execution.

Discrete particle methods

In this section:

- *Discrete element method analysis*
- *Discrete particle swelling*
- *Particle generator*
- *Lumped kinetic molecular method*

Discrete element method analysis

Products: Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

PD3D

Features tested

Contact interaction between:

- two discrete particle elements;
- a discrete particle element and a rigid plane; and
- JKR and shifted JKR force-displacement relation between contacting discrete particles.

Problem description

This section provides basic verification tests for nonadhesive contact and adhesive contact between discrete particles. For the nonadhesive contact the normal and tangential contact formulations in Abaqus/Explicit are compared with the analytical results based on the Hertzian contact formulation with friction in five tests described in *Chung (2011)*. For the adhesive contact case the results for the JKR and shifted JKR adhesion models in Abaqus/Explicit are compared with analytical solutions.

Table 1: Seven types of tests to verify normal and tangential contact formulation for discrete particle elements.

	Description of contact	Feature tested
Test 1	Elastic head-on collision of two identical spheres	Elastic head-on contact between two spheres
Test 2	Elastic normal impact of a sphere with a rigid plane	Elastic normal contact between a sphere and a plane
Test 3	Oblique impact of a sphere with a rigid plane at a constant normal velocity and an incident angle	Tangential contact between a sphere and a plane
Test 4	Head-on collision of two identical spheres at the same translational speed but with equal and opposite angular speed	Tangential contact between two spheres
Test 5	Head-on collision of two different spheres with different translational and angular velocities	Tangential contact between two spheres
Test 6	Normal adhesive contact between two particles	JKR adhesion between two spheres.
Test 7	Normal adhesive contact between two particles	Shifted JKR adhesion between two spheres.

These seven tests characterize different impact scenarios between discrete particle elements and a discrete particle element with a rigid plane.

Model:

Test 1

A head-on collision of two identical spheres of radii 0.01 m with equal and opposite translational speed.

Test 2

A collision of a sphere of radius 0.1 m with translational velocity and a fixed rigid plane in the normal direction.

Test 3

An impact between a fixed rigid plane and a sphere of radius 1.00×10^{-5} m at a constant normal velocity and varying incident angles. This test involves a series of simulations, each with a different tangential velocity of the sphere to characterize a particular incident angle.

Test 4

A head-on collision of two identical spheres of radii 0.1 m with the same translational speed but with equal and opposite angular speed. This test involves a series of simulations, each with a different angular speed of the spheres.

Test 5

A head-on collision of two differently sized spheres with different translational and angular velocities. A sphere of radius 0.1 m has translational and angular velocity; the other sphere, which is five times bigger and 1000 times denser, is initially stationary. The test has multiple simulations, each with a different angular velocity of the smaller sphere.

Test 6

Normal contact between two spheres of the same size. A sphere of radius 5.0 mm is brought into contact with another fixed sphere of the same radius. The spheres are compressed against each other and are then separated. The motion of the first sphere is controlled via a displacement-type boundary condition. The JKR adhesive contact interface is specified between the contact particles.

Test 7

Normal contact between two spheres of the same size. A sphere of radius 5.0 mm is brought into contact with another fixed sphere of the same radius. The spheres are compressed against each other and are then separated. The motion of the first sphere is controlled via a displacement-type boundary condition. The shifted JKR adhesive contact interface is specified between the contact particles.

Mesh: The spheres in all tests are modeled using discrete particle elements (PD3D), and the rigid plane (if applicable) is modeled using conventional shell elements (S4R) that are rendered rigid.

Material: The seven tests are all conducted for two different materials, as described in *Table 2*.

Table 2: Material properties of the spheres.

	Property	Young's Modulus (GPa)	Poisson's ratio	Density (kg/m^3)
Test 1	Glass	48.0	0.20	2800
	Limestone	20.0	0.25	2500
Test 2	Aluminum alloy	70.0	0.30	2699
	Magnesium alloy	40.0	0.35	1800
Test 3	Steel	208	0.30	7850
	Polyethylene	1.0	0.40	1400
Test 4	Aluminum alloy	70.0	0.33	2700
	Copper	120	0.35	8900
Test 5	Aluminum alloy	70	0.33	2700*
	Nylon	2.5	0.40	1000*
Test 6	Polyethylene	1.0	0.42	1400
Test 7	Polyethylene	1.0	0.42	1400

*Density of the smaller sphere. The bigger sphere is 1000 times denser.

Boundary conditions: Wherever applicable, the rigid plane is fixed in all degrees of freedom.

Initial conditions: In tests 1 through 5 the spheres are given an initial translational and angular velocity (if applicable), as described in *Table 3*. The spheres in tests 6 and 7 do not have any initial conditions. The motion of the spheres is specified using displacement-type boundary conditions using an amplitude curve.

Table 3: Initial velocities of the spheres.

	Initial translational speed (m/s)	Initial angular speed (rad/s)
Test 1	10.0	—
Test 2	0.2	—
Test 3	5.0	0.1, 0.2, 0.4, 0.8, 1.5, 3.0, 5.0, 8.0, 10.0, 11.0, 12.0, 20.0
Test 4	0.2	0.175, 0.4, 0.8, 1.5, 3.0, 5.0, 10.0, 20.0
Test 5*	0.2	0.175, 0.25, 0.4, 0.8, 1.5, 3.0, 4.0, 5.0, 6.0, 8.0, 10.0, 20.0
Test 6	0.004 in compression and 0.001 in tension via a smooth amplitude curve	0
Test 7	0.004 in compression and 0.001 in tension via a smooth amplitude curve	0

*Initial speed of the smaller sphere. The bigger sphere is stationary.

Contact formulation: In tests 1–5 the general contact formulation in Abaqus/Explicit is used. Contact is enforced using the tabular pressure-overclosure relationship. Since the contact area of the discrete element is unity, the pressure-overclosure relationship is actually given as force-penetration data computed through Hertzian contact relations with friction:

$$F = \frac{4}{3} E \sqrt{R} \sqrt{\delta^3},$$

where

$$R = \frac{R_1 R_2}{R_1 + R_2},$$

$$\frac{1}{E} = \frac{1-\nu_1^2}{E_1} + \frac{1-\nu_2^2}{E_2},$$

F is the contact force, δ is the penetration, R_1 and R_2 are the radii of the two spheres, E_1 and E_2 are the Young's moduli, and ν_1 and ν_2 are the Poisson's ratios of the materials of two spheres. A rigid plane is approximated by using a large radius and Young's moduli for one of the spheres in the Hertzian contact relations.

For tests 6 and 7 the JKR and shifted JKR adhesive contact interface model is specified between the contact spheres. The surface energy between contacting spheres for both these tests is 50.0 J/m^2.

The friction coefficient for tangential contact behavior for each test is defined as listed in *Table 4*.

Table 4: Friction coefficients.

	Friction coefficient
Test 1	0.35
Test 2	0.00
Test 3	0.30
Test 4	0.40
Test 5	0.40
Test 6	0.0
Test 7	0.0

Contact damping is absent in all tests; hence, the coefficient of restitution of the collision in the normal direction is 1.0, while friction is the only source of energy dissipation.

Results and discussion

Test 1

The elastic contact force is plotted against penetration (see *Figure 1*) for the two materials and for the analytical results. The maximum contact force, the maximum penetration, and the duration of contact are compared with the analytical results (see *Chung, 2011*), as shown in *Table 5*.

Table 5: Normal collision of two spheres.

Property	Glass		Limestone	
	Abaqus	Analytical	Abaqus	Analytical
Contact duration (μs)	40.85	40.34	54.65	54.20
Maximum penetration (μm)	274.87	274.11	369.06	368.30
Maximum contact force (N)	10741.3	10696.9	7130.2	7108.1

Test 2

The elastic contact force is plotted against penetration (see *Figure 2*) for the two materials and for the analytical results. The maximum contact force, the maximum penetration, and the duration of contact are compared with the analytical results (see *Chung, 2011*), as shown in *Table 6*.

Table 6: Normal collision of a sphere with a rigid plane.

Property	Aluminum alloy		Magnesium alloy	
	Abaqus	Analytical	Abaqus	Analytical
Contact duration (μs)	732	731.59	767	766.99
Maximum penetration (μm)	49.71	49.72	52.12	52.12
Maximum contact force (N)	11369.4	11370.8	7232.1	7232.0

Test 3

The normalized incident angle, the normalized recoil angle, and the normalized postcollision angular speed for the sphere are calculated for each simulation. The normalized incident angle is the ratio of the precollision relative tangential velocity of the contact point of the spheres to the precollision relative normal velocity of the center of the spheres scaled by the friction coefficient. The normalized postcollision angular speed is the ratio of the postcollision angular speed scaled by the radius of the sphere to the postcollision normal velocity of the sphere. *Figure 3* shows a plot of the normalized recoil angle versus the normalized incident angle compared with the analytical results. The normalized recoil angle is the ratio of the postcollision relative tangential velocity of the contact point of the spheres to the postcollision relative normal velocity of the center of the spheres scaled by the friction coefficient. *Figure 4* shows a plot of the normalized postcollision angular speed versus the normalized incident angle compared with the analytical results. The plots in *Figure 3* and *Figure 4* show that sticking persists until the initial tangential velocity reaches a threshold value, beyond which slipping occurs during the collision.

Test 4

Figure 5 shows a plot of the calculated postcollision tangential speed of the contact point versus the initial angular speed of the sphere compared with the analytical results. In *Figure 6* the postcollision angular speed is plotted against the initial angular speed of the sphere along with the analytical results. Since the angular speed is the same for both the spheres with opposite direction of spin, no relative slip occurs during the collision; hence, the postcollision tangential velocity is absent. The comparison between the initial and the final angular speeds shows that there is no energy dissipation due to friction in this model because relative slip is absent.

Test 5

The tangent of the incident and the recoil angles are evaluated for each of the simulations. The recoil angle is the ratio of the relative tangential velocity of the contact point of the spheres to the relative normal velocity of the center of the spheres after the collision. The incident angle is the same quantity evaluated before the collision. *Figure 7* shows a plot of the tangent of the recoil angle against that of the incident angle. As in test 3, below a threshold angular velocity, the smaller sphere sticks to the bigger sphere. Beyond this threshold value, it slips during the collision.

Test 6

The blue curve in *Figure 8* shows the force-displacement response. The analytical value of the pull-off force is −0.589 N. The analytical value of the separation distance is −0.0044174 mm. Both of these values agree with the numerical results obatined from Abaqus/Explicit.

Test 7

The red curve in *Figure 8* shows the force-displacement response. The analytical value of the pull-off force is −0.589 N. This is the same value as for the unshifted JKR adhesion model. The analytical value of the separation distance is −0.0097687 mm. Both of these values agree with the numerical results obtained from Abaqus/Explicit.

Input files

normal_identical_part_part.inp	Test 1: elastic head-on collision of two identical spheres.
normal_part_face.inp	Test 2: elastic normal impact of a sphere with a rigid plane.
oblique_part_face.inp	Test 3: oblique impact of a sphere with a rigid plane at a constant normal velocity and an incident angle.
normal_oppspin_identical_part_part.inp	Test 4: head-on collision of two identical spheres at the same translational speed, but with equal and opposite angular speed.
normal_spin_part_part.inp	Test 5: head-on collision of two different spheres with different translational and angular velocities.
part_part_jkr_jkrshift.inp	Test 6 and 7: normal contact between two spheres with JKR adhesion and shifted JKR adhesion models, respectively.
normal_identical_part_part_autodt.inp	Elastic head-on collision of two identical spheres using automatic time incrementation.
normal_part_face_autodt.inp	Elastic normal impact of a sphere with a rigid plane using automatic time incrementation.
oblique_part_face_autodt.inp	Oblique impact of a sphere with a rigid plane at a constant normal velocity and an incident angle using automatic time incrementation.
normal_oppspin_identical_part_part_autodt.inp	Head-on collision of two identical spheres at the same translational speed, but with equal and opposite angular speed using automatic time incrementation.
normal_spin_part_part_autodt.inp	Head-on collision of two different spheres with different translational and angular velocities using automatic time incrementation.
part_part_jkr_jkrshift_autodt.inp	Normal contact between two spheres with JKR adhesion and shifted JKR adhesion models, respectively, using automatic time incrementation.

References

- Chung, Y. C., and J. Y. Ooi, “Benchmark Tests for Verifying Discrete Element Modelling Codes at Particle Impact Level,” *Granular Matter*, vol. 13, pp. 643–656, 2011.

Figures

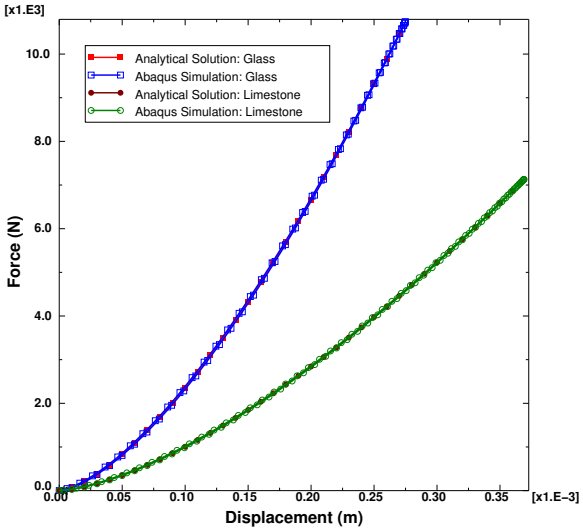


Figure 1: Contact force vs. penetration for collision of two spheres.

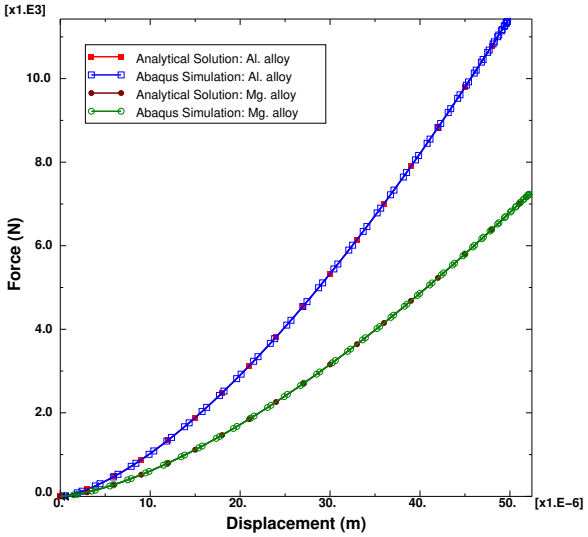


Figure 2: Contact force vs. penetration for collision of a sphere with a rigid plane.

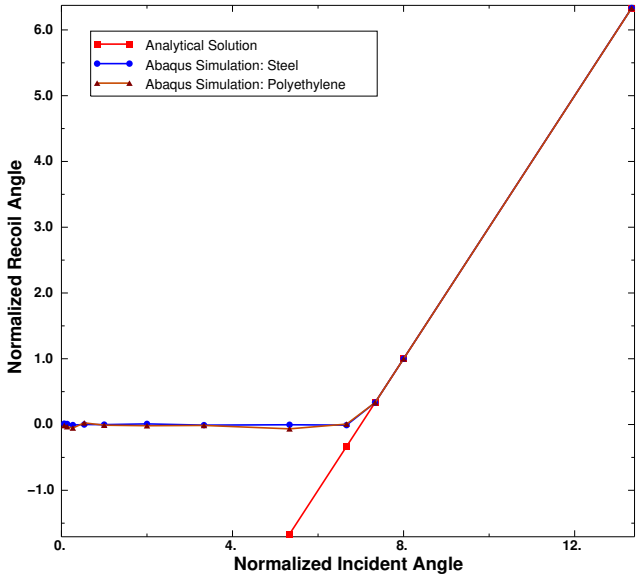


Figure 3: Normalized recoil angle vs. normalized incident angle.

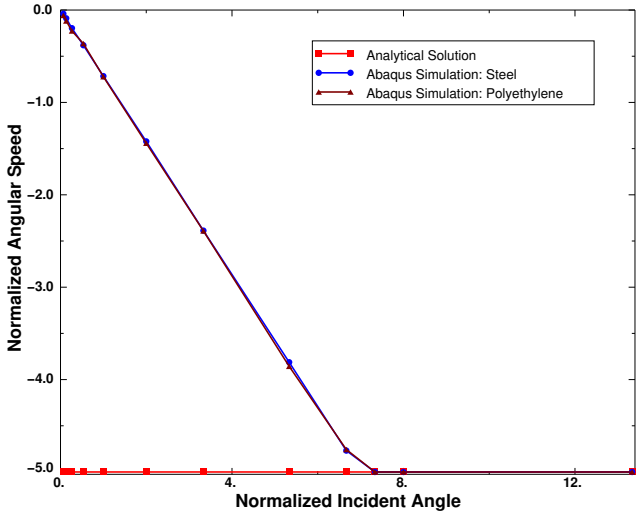


Figure 4: Normalized postcollision angular speed vs. normalized incident angle.

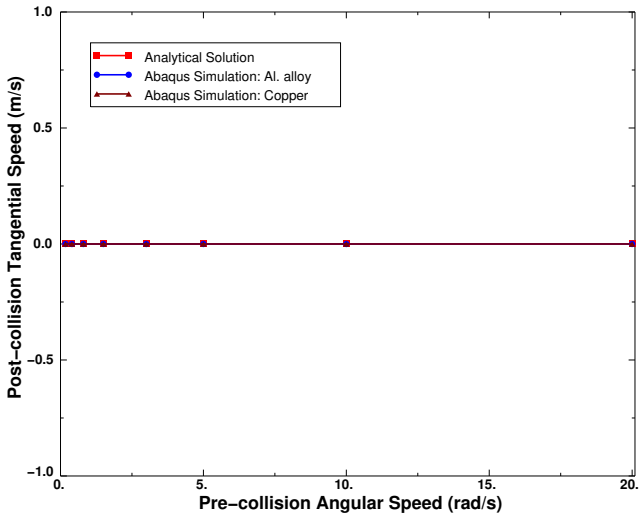


Figure 5: Postcollision tangential speed vs. precollision angular speed.

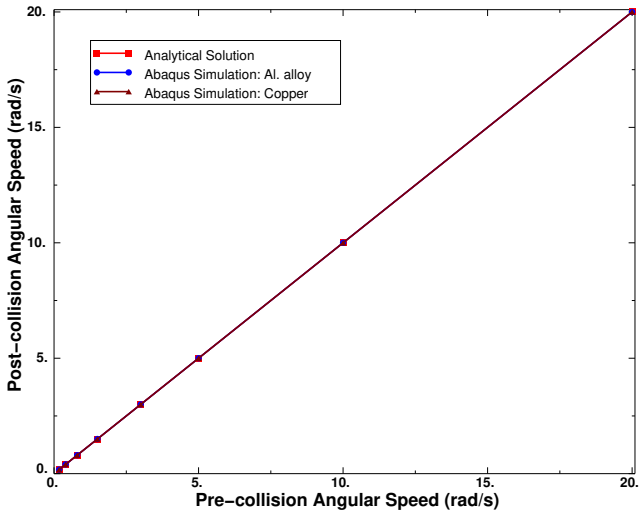


Figure 6: Postcollision angular speed vs. precollision angular speed.

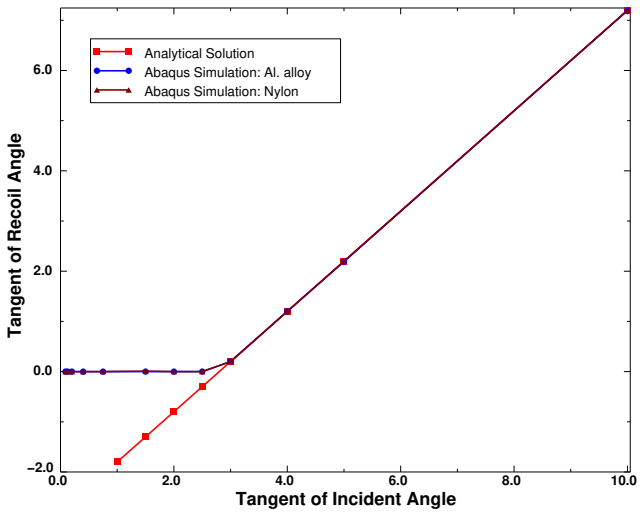


Figure 7: Tangent of the recoil angle vs. tangent of the incident angle of the smaller sphere.

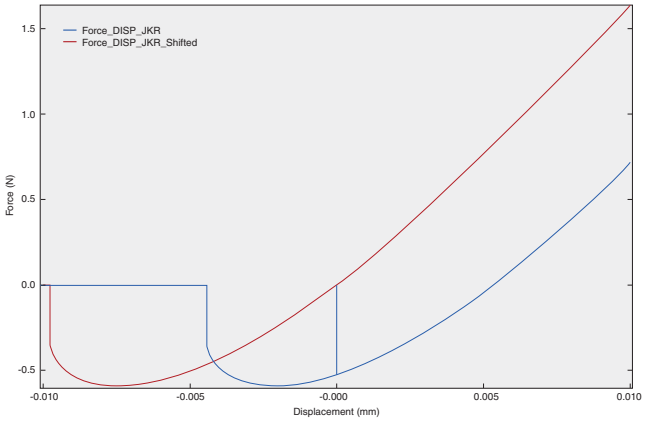


Figure 8: Force vs. displacement relationship for JKR and shifted JKR adhesion model.

Discrete particle swelling

Products: Abaqus/Explicit

References:

- *Discrete Element Method*

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Features tested

This section provides basic verification tests for discrete particle swelling.

Problem description

In this verification problem the evolution of the particle radius is verified against a first-order model that is fitted to experimental data.

Model: When a super absorbent polymer (SAP) absorbs fluid, it undergoes a large increase in size. This test consists of two SAP particles of the same size. A first-order swelling model as described in *Braile et al. (2022)* is used to construct the particle swelling table in Abaqus/Explicit. *Braile et al. (2022)* describes a first-order model where the evolution of a dimensionless swelling parameter v' as a function of time t is given by $v' = 1 - e^{-kt}$. Here k is a kinetic parameter that is obtained from fitting the model to experimental data. The swelling parameter $v' = \frac{v - v_0}{v_{eq} - v_0}$, where v, v_0, and v_{eq} are the current, initial, and equilibrium volumes, respectively.

SAP particle properties: All properties of the SAP particles used for this test are obtained from *Braile et al. (2022)*. From the distribution provided in *Braile et al. (2022)*, the initial radius of the particles is taken as 1.5×10^{-4} m. This translates to an initial volume v_0 of $1.41372 \times 10^{-11} m^3$. The particle swells to an equilibrium volume v_{eq} of $2.78 \times 10^{-8} m^3$. The kinetic parameter k for the SAP particles obtained from experimental data is 3×10^{-3} 1/sec. The above properties are used to compute the volume change ratio for the Abaqus/Explicit model. The computed data for the volumetric particle swelling definition is shown in *Table 1*.

Table 1: Particle swelling data.

Volume change ratio	Temperature
0.0	0.0
1956.571	1.0

The volume change ratio is modeled as a function of temperature only. No permanent surface deposition is applied in this test.

Applying temperature to swell particles: To swell the particles, the temperature is increased during the analysis. The amplitude curve to vary the temperature as a function of time is the same as the dimensionless swelling parameter in the first-order model presented in *Braile et al. (2022)*.

Results and discussion

The swelling based on the volume change ratio driven by temperature occurs over 1800 seconds. *Figure 1* compares the history of the particle radius from Abaqus/Explicit against the particle radius computed from the first-order model. The evolution of the particle radius over time from Abaqus/Explicit shows good agreement with that computed in *Braile et al. (2022)*.

Input files

particle_swelling_ver.inp	Particle swelling based on a first-order model.

References

- Braile, D., C. Hare, and C. Y. Wu, DEM Analysis of Swelling Behaviour in Granular Media, *Advanced Powder Technology*, vol. 33, 2022.

Figures

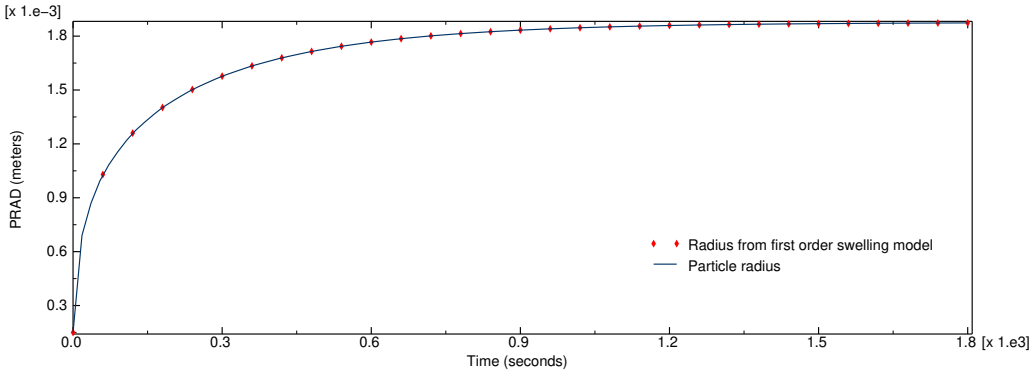


Figure 1: Comparison of particle radius.

Particle generator

Products: Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

PD3D

Features tested

- Probability density of generator particles
- Mass flow rate per inlet area of generated particles

Problem description

The probability density of generator particles and the mass flow rate per inlet area of generated particles are tested for four particle generators, each generating a single particle species. Each particle generator has a single faceted inlet surface with constant dimensions of 7 mm × 7 mm.

Table 1 lists the particle size probability density function (PDF) type and parameters for each particle generator species. The mass flow rate per unit inlet area for each particle generator species is held constant for the duration of the analysis. Particles of all species have a constant entry speed of 1000 mm/s. All generated particles in this problem continue to travel with the entry velocity along the normal direction to the generating inlet facet. Each particle generator can generate a maximum of 20000 particles. *Figure 1* presents a view that is normal to the inlet facets showing the generated particles at the end of the analysis.

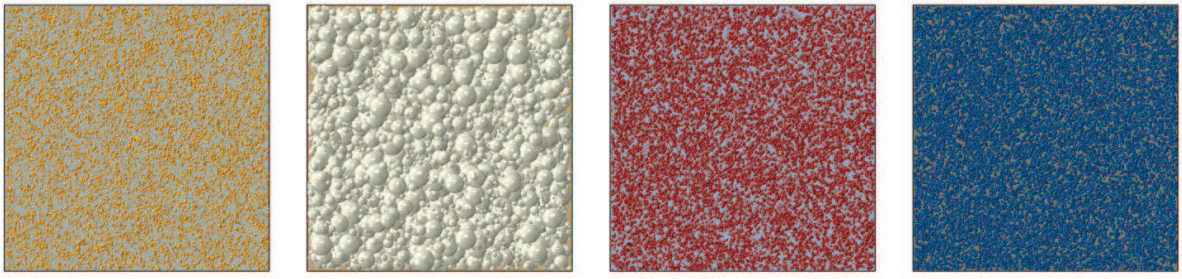


Figure 1: Four particle generators.

Table 1: Particle generator description.

Particle generator	Probability density function	Particle size distribution	Mass flow rate per unit inlet area
PG1	Normal distribution	Mean = 0.01 mm Standard deviation = 0.02 mm	5.0×10^{-8} t/mm^{2}s for 0.02 s

Particle generator	Probability density function	Particle size distribution	Mass flow rate per unit inlet area
		Minimum radius = 0.0005 mm	
		Maximum radius = 0.05 mm	
PG2	Log-normal distribution	Mean = 0.25 mm	5.0×10^{-8} t/mm^{2}s for 0.02 s
		Standard deviation = 1.0 mm	
		Minimum radius = 0.0005 mm	
		Maximum radius = 0.3 mm	
PG3	Uniform distribution	Minimum radius = 0.0005 mm	5.0×10^{-8} t/mm^{2}s for 0.02 s
		Maximum radius = 0.05 mm	
PG4	Piecewise linear distribution	Minimum radius = 0.0001 mm	5.0×10^{-8} t/mm^{2}s for 0.02 s
		Maximum radius = 0.057 mm	

Results and discussion

A direct user time increment of 1.0×10^{-4} s was specified for the analysis. Generated and specified PDFs and mass flow rates are compared for particle generators PG1, PG2, PG3, and PG4.

Comparison of generated PDF

The specified normal and the log-normal probability density functions for particle generators PG1 and PG2, respectively, are truncated PDFs. Truncated analytical PDF curves are compared to the generated results for these two cases (see *Probability Density Function* for more details on truncated PDFs). *Figure 2* and *Figure 3* show the comparison of generated PDFs with the specified truncated PDF for generators PG1 and PG2, respectively. *Figure 4* and *Figure 5* show the comparison of generated uniform and piecewise linear PDFs with the corresponding specified PDFs for generators PG3 and PG4, respectively.

The two main factors influencing the agreement between the generated and specified PDF are the rejection of candidate particle sizes during the generation process (see *Particle Generator* for further details) and the number of particles being generated. The figures show that the generated PDF follows the specified PDF in an average sense in all four cases.

Comparison of generated mass flow rate

Since the inlet area does not change, the mass flow rate (MFR) for this problem is

$$5.0 \times 10^{-8}\ \mathrm{t/mm^2 s} \times (7\ \mathrm{mm} \times 7\ \mathrm{mm}) = 2.45 \times 10^{-6}\ \mathrm{t/s}.$$

Figure 6, *Figure 8*, and *Figure 9* show the comparison of achieved to specified mass flow rates for particle generators PG1, PG3, and PG4, respectively. Particle generators PG1, PG3, and PG4 exhaust the maximum number (20000) of particles and halt generating particles, causing the mass flow rates for these generators to drop to zero at 0.004 s, 0.0085 s, and 0.0138 s, respectively. The achieved mass flow rate closely matched the specified mass flow rate for these three particle generators. Particle generator PG2 generates 8048 particles during the analysis. *Figure 7* shows the comparison of achieved to specified mass flow rate for particle generator PG2. Due to the discrete nature of particle generation and larger particle sizes, the achieved mass flow rate matched the specified mass flow rate only in an average sense for particle generator PG2 (see *Particle Generator* for further details).

Input files

pg4_species1_alldist_cmfr.inp	Verify PDF and mass flow rate for particle generator.

Figures

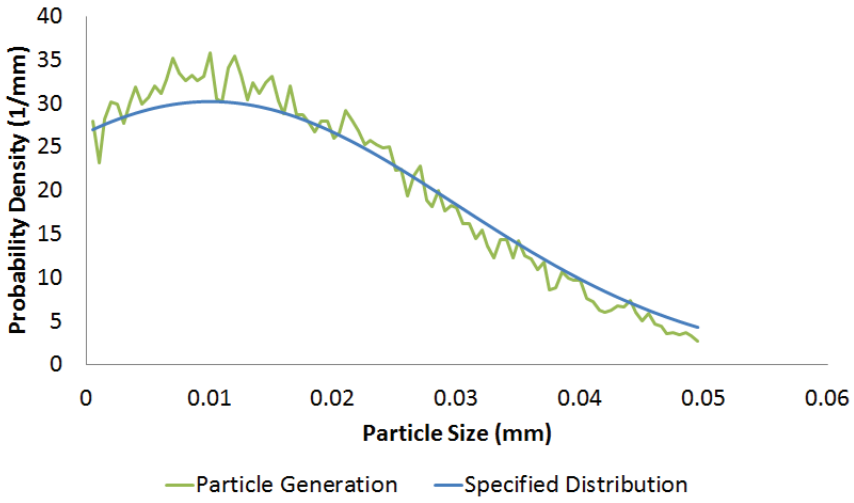


Figure 2: Comparison of generated vs. specified PDF for particle generator 1.

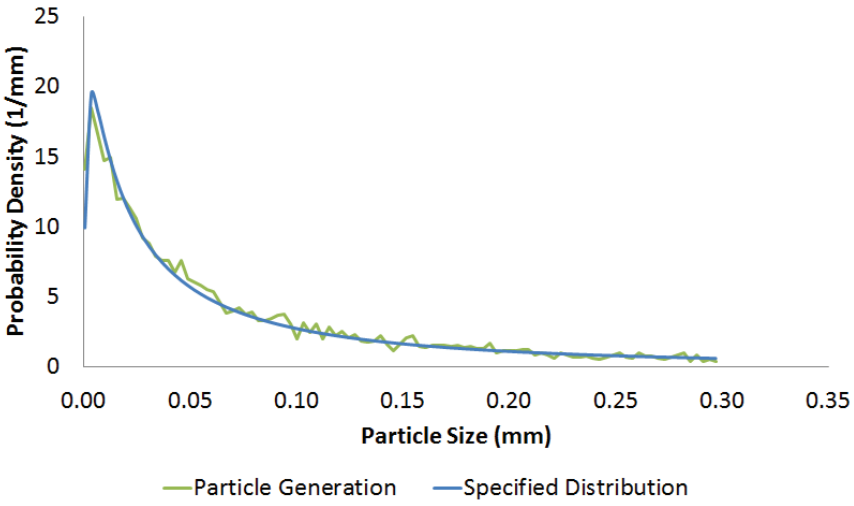


Figure 3: Comparison of generated vs. specified PDF for particle generator 2.

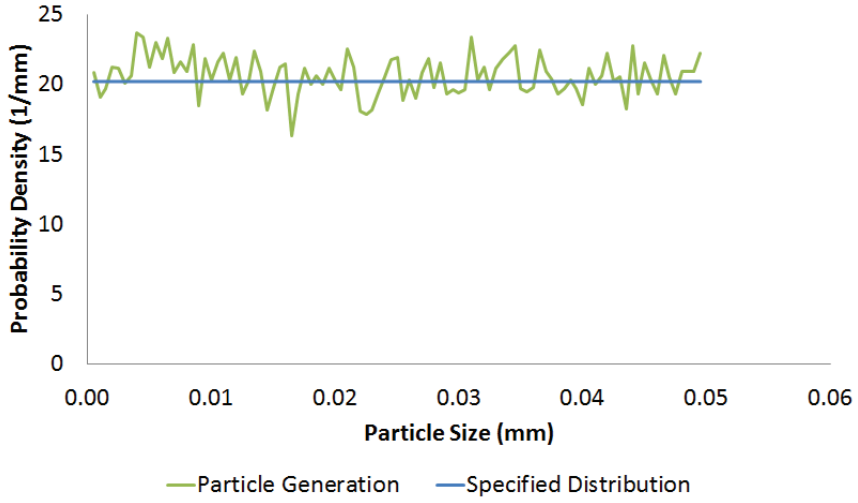


Figure 4: Comparison of generated vs. specified PDF for particle generator 3.

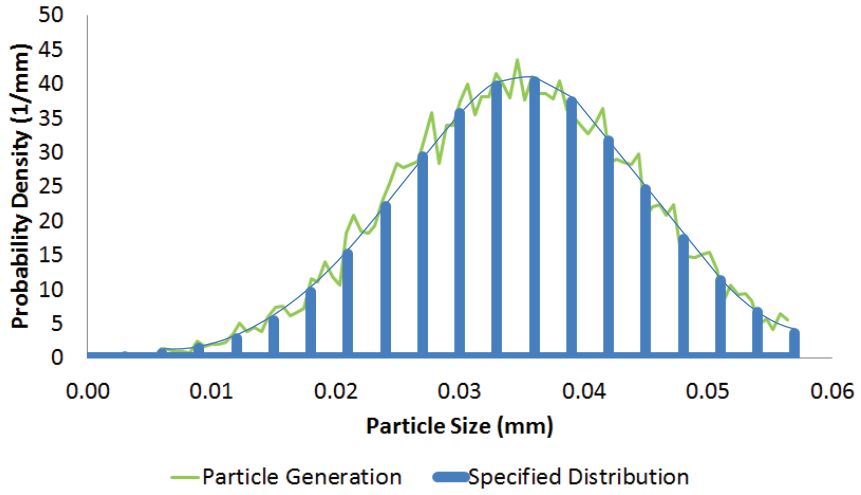


Figure 5: Comparison of generated vs. specified PDF for particle generator 4.

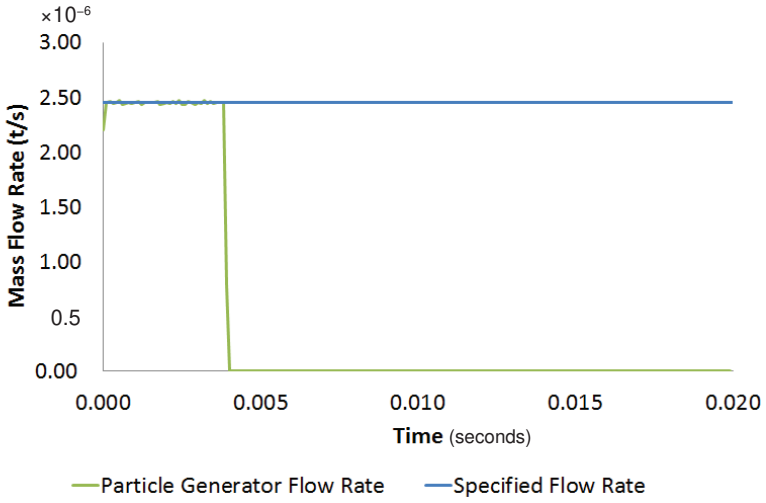


Figure 6: Comparison of achieved vs. specified MFR for particle generator 1.

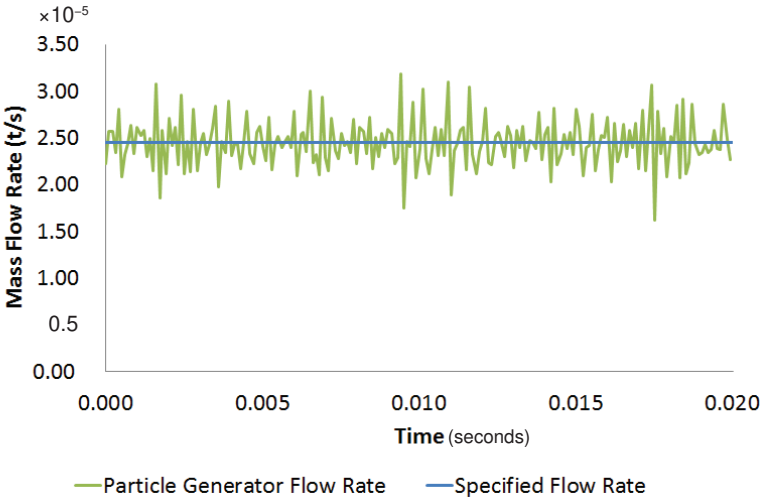


Figure 7: Comparison of achieved vs. specified MFR for particle generator 2.

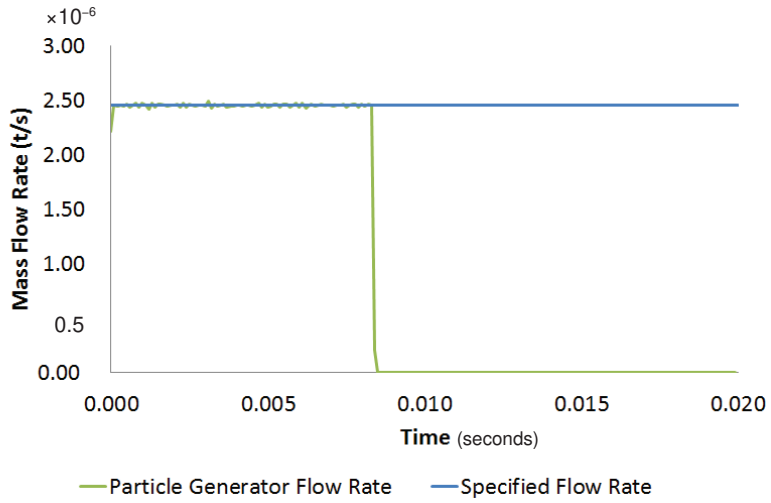


Figure 8: Comparison of achieved vs. specified MFR for particle generator 3.

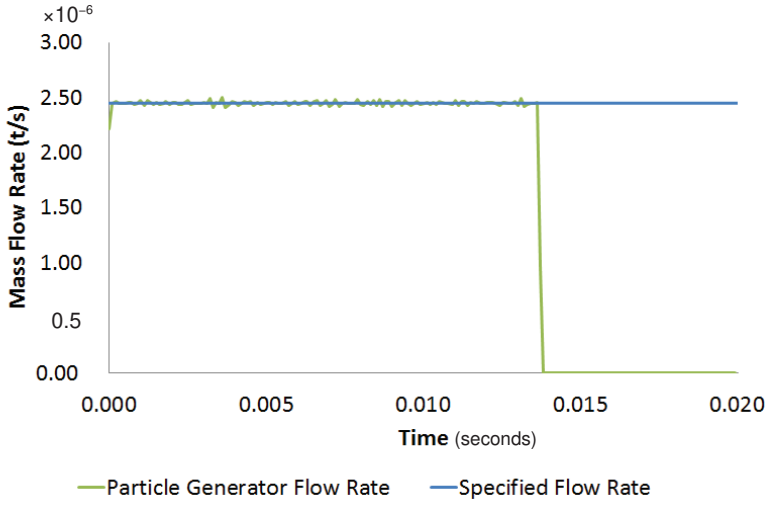


Figure 9: Comparison of achieved vs. specified MFR for particle generator 4.

Lumped kinetic molecular method

Products: Abaqus/Explicit

References:

• *Lumped Kinetic Molecular Method*

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Features tested

This section provides basic verification tests to define:

- Particle generators
- Fluid cavities
- Discrete sections
- Gas species used for particle generators
- Fluid inflator properties used with particle generators

Problem description

This verification problem uses the lumped kinetic molecular (LKM) method. The mass, average pressure, and average temperature of the generated gas are verified against the uniform pressure method (UPM).

Model	The tank is in the shape of a rectangular box and is meshed using membrane elements. Boundary conditions are applied to all the nodes to make the tank rigid. A particle generator is defined inside the box. The inlet of the particle generator is positioned close to one end of the tank. The tank is initially empty.
Gas material	The particle generator generates gas particles with properties of air. The molecular weight of the gas is 2.897×10^{-6} tonnes/mol, and its heat capacity at constant pressure is 31829.339 J/K mol. To model 2×10^{-7} tonnes of the gas, 200,000 lumped particles are used. The gas is generated at a constant rate of 4×10^{-5} tonnes/sec at a temperature of 100K. The particle generator stops generating at 0.005 seconds.
Boundary conditions	The tank is held rigid until 0.006 seconds. After the gas reaches an equilibrium state, the wall behind the particle generator inlet is held fixed while the side walls are stretched such that the final volume of the tank is 1.5 times its original volume. The stretching of the box stops at 0.009 seconds, and the volume of the tank stays constant to the end of the analysis.

Results and discussion

The generated gas undergoes the following four phases during this test:

- the initial generation phase,
- a static pressure phase,
- an adiabatic expansion phase,
- and the final static equilibrium phase.

The generated gas reaches an equilibrium state around 0.006 seconds. As the tank is stretched, the gas undergoes adiabatic expansion. Since the mass of the gas stays unchanged during the expansion phase, the pressure and temperature both decrease. The gas once again reaches an equilibrium state after expansion stops at 0.009 seconds.

Figure 1 compares the generated gas mass for the LKM method with the generated mass using the UPM. The four phases in this test can be seen in *Figure 2*, which compares the average gas pressure in the LKM analysis with the gas pressure obtained from a corresponding UPM analysis. The results from the two methods show close agreement. *Figure 3* compares the average gas temperature from the LKM solution with that obtained from the UPM.

Input files

ae_lkm_tank_test.inp	Static pressure followed by adiabatic tank test.

Figures

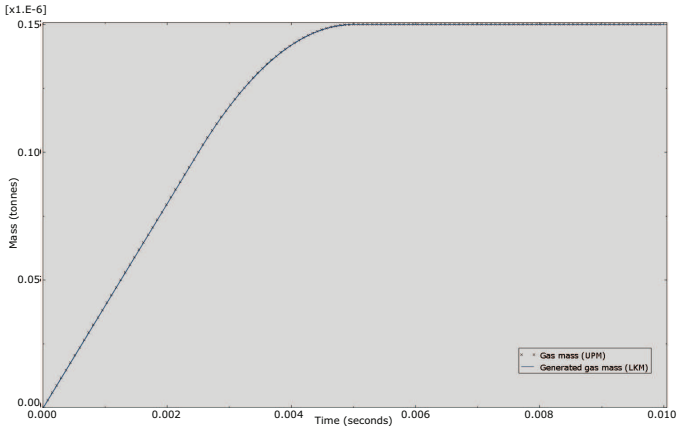


Figure 1: Comparison of generated gas mass.

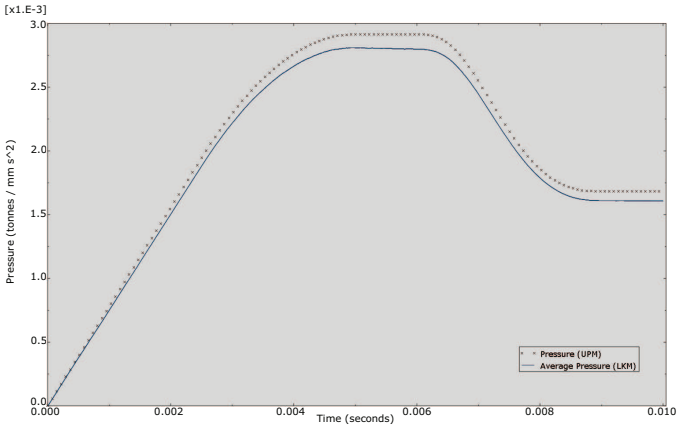


Figure 2: Comparison of average gas pressure.

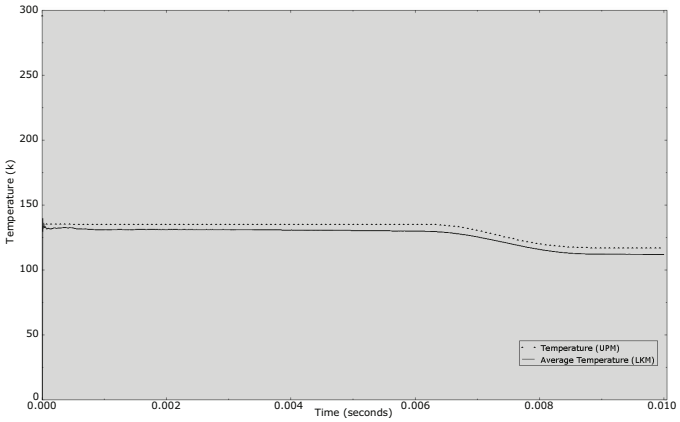


Figure 3: Comparison of average gas temperature.

Miscellaneous procedures and techniques

In this section:

- *Direct cyclic and low-cycle fatigue analyses*
- *Complex eigenvalue extraction*
- *CEL analysis of a rotating water disk*
- *Smoothed particle hydrodynamic analysis*
- *Frequency extraction using the AMS eigensolver*
- *Steady-state dynamics with nondiagonal damping using the AMS eigensolver*
- *Media transport*
- *Transferring results between dissimilar meshes in Abaqus/Standard*
- *One-step inverse analysis*
- *Ramping of initial stress*
- *Undrained pore fluid flow and stress analysis*
- *Controlling material rate dependence*

Direct cyclic and low-cycle fatigue analyses

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

The tests in this section verify the direct cyclic analysis procedure and the low-cycle fatigue procedure using the direct cyclic approach for structures subjected to different types of cyclic loadings, which include distributed forces, concentrated forces, displacements, and temperatures. The direct cyclic and low-cycle fatigue procedures are also verified when they are preceded or followed by other procedures in a single analysis or in a restart analysis.

A simple cube

Elements tested

C3D8 C3D10

Features tested

A simple cube subjected to different cyclic loadings.

Problem description

The model in each test consists of twelve tetrahedral elements or one brick element. All the nodes at one end ($z = 0$) are constrained along the z-axis. Cyclic distributed loads, concentrated loads, or displacements are applied in the z-direction to the nodes at the other end ($z = 1$). Both kinematic hardening plasticity models and two-layer viscoplasticity models are used.

Results and discussion

The results obtained using the direct cyclic procedure are compared with those obtained using the classical approach, which involves applying cyclic loadings repetitively to the model in multiple steps using the static or quasi-static procedures. The shapes of the stress-strain curves in a stabilized cycle obtained using both approaches are consistent.

Input files

dircyclic_cload_ffouri_ftinc.inp	Cyclic concentrated loadings with fixed number of Fourier terms and fixed time incrementation.
dircyclic_cload_ffouri_ftinctp.inp	Cyclic concentrated loadings with fixed number of Fourier terms and fixed time incrementation used with the **TIME POINTS* option.
dircyclic_cload_vfouri_ftinc.inp	Cyclic concentrated loadings with varying number of Fourier terms and fixed time incrementation.
dircyclic_cload_vfouri_ftinctp.inp	Cyclic concentrated loadings with varying number of Fourier terms and fixed time incrementation used with the **TIME POINTS* option.
dircyclic_cload_ffouri_vtinctp.inp	Cyclic concentrated loadings with fixed number of Fourier terms and automatic time incrementation used with the **TIME POINTS* option.
dircyclic_precload.inp	Static pre-loading step.

dircyclic_cload_ffouri_ftinc_r.inp	Restart of dircyclic_precload.inp. Cyclic concentrated loadings with fixed number of Fourier terms and fixed time incrementation.
dircyclic_cload_ffouri_ftinc_rs.inp	Restart of dircyclic_cload_ffouri_ftinc_r.inp. Cyclic concentrated loadings with fixed number of Fourier terms and fixed time incrementation.
dircyclic_cload_ffouri_ftinc_ps.inp	Post output of dircyclic_cload_ffouri_ftinc_r.inp.
dircyclic_cload_ffouri_ftinc_ms.inp	Multiple direct cyclic analysis steps in a single analysis. Cyclic concentrated loadings with fixed number of Fourier terms and fixed time incrementation.
dircyclic_dload_ffouri_ftinc.inp	Cyclic distributed loadings with fixed number of Fourier terms and fixed time incrementation.
dircyclic_disp_ffouri_ftinc.inp	Cyclic displacement loadings with fixed number of Fourier terms and fixed time incrementation.
dircyclic_cloadc_vfouri_ftinc.inp	A general static step with contact followed by a direct cyclic step involving cyclic concentrated loadings with varying number of Fourier terms and fixed time incrementation.

A simple sheet with a circular hole

Elements tested

CPE4R

Features tested

A simple sheet with a circular hole subjected to different cyclic loadings.

Problem description

The undeformed square sheet is 1.5 mm thick and is 7.5 mm on each side. It has a centrally located internal hole of radius 0.25 mm. The body is modeled with 128 plane strain reduced-integration elements (element type CPE4R). The symmetry conditions at $\boldsymbol{x = 0}$ and at $\boldsymbol{y = 0}$ are imposed with a boundary condition. The edges parallel to the *x*-axis are restrained from stretching in the *y*-direction. Cyclic concentrated forces or cyclic distributed forces are imposed on the right-hand edge of the mesh in the *x*-direction. For the case where cyclic thermal loadings read from the results file of a heat transfer analysis are imposed, the right-hand edge is also constrained in the *x*-direction. Both kinematic hardening plasticity models and two-layer viscoplasticity models are used.

Results and discussion

The results (stress-strain curves) obtained using the direct cyclic procedure are compared with those obtained using the classical approach, which involves applying cyclic loadings repetitively to the model in multiple steps using the static or quasi-static procedures. The shapes of the stress-strain curves in a stabilized cycle obtained using both approaches are consistent. In the case where cyclic concentrated forces are applied to the model, plastic ratcheting occurs in which the shape of the stress-strain curve does not change but the mean value of the strains keeps shifting. This behavior is predicted by using both the direct cyclic approach and the classical approach.

Input files

dircyclic_heat.inp	Heat transfer analysis.
dircyclic_temp_ffouri_ftinc.inp	Cyclic thermal loadings with temperatures read from the results file of the heat transfer run (dircyclic_heat.inp).

dircyclic_rtemp_vfouri_ftinc.inp	Cyclic thermal loadings with temperatures read from the results file of the heat transfer run (dircyclic_heat.inp) and ramped up to their initial condition values.
dircyclic_dload_vfouri_ftinc.inp	Cyclic distributed loadings with varying number of Fourier terms and fixed time incrementation.
dircyclic_cload_vfouri_vtinctp.inp	Cyclic concentrated loadings with varying number of Fourier terms and automatic time incrementation used with the **TIME POINTS* option.
dircyclic_cload_vfouri_vtinc_ps.inp	Post output of dircyclic_cload_vfouri_vtinctp.inp.

A round notch bar

Elements tested

CAX4

Features tested

A round notch bar subjected to a cyclic loading.

Problem description

The undeformed round notch bar is 75 mm long, with a 2 mm notch radius and a section diameter of 10 mm. The body is modeled with 672 4-node bilinear axisymmetric quadrilateral elements (element type CAX4). The symmetry conditions at $y = 0$ are imposed with a boundary condition. The edges parallel to the x-axis are subjected to displacement loadings in the y-direction. A static step with a displacement loading of 0.25 mm is followed by a low-cycle fatigue step. A sinusoidal cyclic displacement loading between 0.375 mm and 0.125 mm is applied to the low-cycle fatigue step with a time period of 80 seconds. Linear kinematic hardening plasticity model is used.

Results and discussion

The results (scale stiffness degradation, SDEG) obtained using the low-cycle fatigue procedure are compared with those available in the literature (see *Pirondi, 2003*). As the cycling proceeds, damage accumulation at the notch root continues to increase. When the cycle number reaches 50, SDEG is equal to 0.74, similar to the result obtained in *Pirondi (2003)*.

Input files

directcyclic_fatigue_rnb.inp	A static step followed by a low-cycle fatigue step subjected to cyclic displacement loadings.
directcyclic_fatigue_rnb_rest.inp	A low-cycle fatigue step restarted from the low-cycle fatigue step in directcyclic_fatigue_rnb.inp.
directcyclic_fatigue_rnb_rest2.inp	A low-cycle fatigue step restarted from the static step in directcyclic_fatigue_rnb.inp.
directcyclic_fatigue_rnb_ps.inp	Post output of directcyclic_fatigue_rnb.inp.

References

- Pirondi, A., and N. Bonora, “Modeling Ductile Damage under Fully Reversed Cycling,” *Computational Materials Science*, vol. 26, pp. 129–141, 2003.

Complex eigenvalue extraction

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

The tests in this section verify the complex eigenvalue extraction procedure in Abaqus/Standard, which uses the subspace projection method. The procedure is tested for systems with symmetric stiffness matrices that include damping terms and for problems with friction, which introduces unsymmetry to the stiffness matrix.

One-element test

Elements tested

CPE4

Features tested

Complex eigenvalue extraction for a system with a symmetric stiffness matrix, both with and without damping.

Problem description

In both tests the model consists of a quadratic element of unit length. The nodes at one end ($y = 0$) are constrained. The eigenvalue extraction is performed for the undeformed configuration.

Results and discussion

The stiffness matrix in the first problem (pcfreq_ce4sf_real.inp) is symmetric and contains no damping. In the absence of damping contributions, the eigenvalues extracted by the complex eigensolver must have zero real components and the imaginary components (frequencies) must be the same as the frequencies obtained in the preceding frequency extraction step. In the second problem (pcfreq_ce4sf_real.inp) mass-proportional damping is introduced. The following relations can be derived for an underdamped system with mass-proportional damping:

$Re\left(\mu_N\right) = \alpha/2$ and $Im\left(\mu_N\right) = \sqrt{\omega_N^2 - \alpha^2/4}$, where $Re\left(\mu_N\right)$ and $Im\left(\mu_N\right)$ are the real and imaginary components of the complex eigenvalues, respectively; α is the mass-proportional damping factor; and ω_N is the natural frequency of the undamped system. The complex eigenvalues obtained for this problem match the formulae above.

Input files

pcfreq_ce4sf_real.inp	Complex eigenvalue extraction for a symmetric stiffness matrix without damping.
pcfreq_ce4sf_imag.inp	Complex eigenvalue extraction for a symmetric stiffness matrix with mass-proportional damping.

A rotating ring compressed between two plates

Elements tested

C3D8

Features tested

Complex eigenvalue extraction for a system with an unsymmetric stiffness matrix caused by a friction contribution.

Problem description

The model consists of a ring with an inside radius of 1.0 and an outside radius of 2.0 and two plates positioned at both sides of the ring. The ring is modeled using a linear elastic material with a Young's modulus of 200, Poisson's ratio of 0.3, and density of 1.0. Contact pairs define contact between the side surfaces of the ring and the plates. The ring is meshed with 16 linear brick elements (element type C3D8). The plates are modeled with membrane elements (element type M3D4) for the models with deformable-to-deformable contact or with rigid elements (element type R3D4) for the problems with deformable-to-rigid contact.

The loading consists of two steps. In the first step the plates are moved a distance of 0.05 toward the ring to establish frictionless contact. In the second step the friction coefficient is increased to 0.3 and a rotational velocity is imposed on the ring. Because the complex eigensolver uses the subspace projection method, the natural frequencies must be extracted prior to the complex eigenvalue extraction step. The following problems with different contact models are considered:

- deformable-to-deformable contact with small sliding (pcfreq_def_ss.inp),
- deformable-to-deformable contact with small sliding, including friction-induced damping effects (pcfreq_def_ss_fdamp.inp),
- deformable-to-rigid contact with small sliding (pcfreq_rg_ss.inp),
- deformable-to-deformable contact with finite sliding (pcfreq_def_fs.inp), and
- deformable-to-deformable contact with finite sliding in a restarted analysis (pcfreq_def_fs_res.inp).

In addition, analyses with a steady-state transport step (pcfreq_sst_3d.inp), a substructure usage (pcfreq_sup_use.inp), and a velocity-dependent friction coefficient (pcfreq_def_ss_negdamp.inp) are tested.

Results and discussion

An analytic solution is not available for this problem, so the results (the frequency of an unstable mode and the damping ratio) are compared only between the different models. As shown in the table that follows, the results for pcfreq_def_fs.inp, pcfreq_def_fs_res.inp, pcfreq_def_ss.inp, pcfreq_def_ss_fdamp.inp, pcfreq_def_ss_negdamp.inp, pcfreq_rg_ss.inp, and pcfreq_sst_3d.inp are in very good agreement. The differences in the results for pcfreq_sup_use.inp are due to the use of a substructure to model the elastic ring.

Input file	Frequency of an unstable mode	Real part of an unstable mode
pcfreq_def_ss	1.769	0.1648
pcfreq_def_ss_fdamp	1.769	0.1636
pcfreq_rg_ss	1.769	0.1448
pcfreq_def_fs	1.770	0.1650
pcfreq_def_fs_res	1.770	0.1650
pcfreq_sst_3d	1.767	0.1582
pcfreq_sup_use	1.799	0.1487
pcfreq_def_ss_negdamp	1.769	0.1705

Input files

pcfreq_def_ss.inp Deformable-to-deformable contact with small sliding.

pcfreq_def_ss_fdamp.inp	Deformable-to-deformable contact with small sliding, including friction-induced damping effects.
pcfreq_rg_ss.inp	Deformable-to-rigid contact with small sliding.
pcfreq_def_fs.inp	Deformable-to-deformable contact with finite sliding.
pcfreq_def_fs_res.inp	Deformable-to-deformable contact with finite sliding, restarted analysis.
pcfreq_sst_3d.inp	Deformable-to-rigid contact with finite sliding, rotational velocity imposed in the steady-state transport step.
pcfreq_sst_axi.inp	Axisymmetric mesh generation used in pcfreq_sst_3d.inp.
pcfreq_sup_use.inp	Substructure analysis.
pcfreq_sup_gen.inp	Substructure generation file referenced in pcfreq_sup_use.inp.
pcfreq_def_ss_negdamp.inp	Deformable-to-deformable contact with small sliding and velocity-dependent friction coefficient.

CEL analysis of a rotating water disk

Products: Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

EC3D8R

Features tested

Eulerian analysis

Problem description

This example utilizes the pure Eulerian analysis technique to model viscous flow of water between two concentric cylinders.

Model: The model is created in Abaqus/CAE using a simple circular Eulerian domain with an outer radius of 0.07 m and an inner radius of a = 0.04 m (see *Figure 1*). Because Eulerian analyses must be conducted in three-dimensional space, this two-dimensional problem is approximated using a thin domain with a single Eulerian element through its thickness. Rectangular-shaped elements provide the best accuracy and performance in Eulerian analyses, so the thickness is chosen to correspond to the minimum element size.

Mesh: The Eulerian domain is meshed with 160 elements in the circumference and 14 elements along the radial direction (see *Figure 1*). The mesh provides good resolution in the radial direction and reasonable aspect ratio elements. A total of 2240 Eulerian EC3D8R elements are used. The circular (conforming) meshing is employed to avoid the need for Eulerian-Lagrangian contact.

Material: Water is modeled as a nearly incompressible, viscous Newtonian fluid. The linear $U_s - U_p$ Hugoniot form of the Mie-Grüneisen equation of state is used in the material model. The parameters used to define the material are listed in *Table 1*.

Boundary conditions: To approximate the rotation of the water disk, the water is subjected to a uniform tangential velocity of $U = 0.2932$ m/s at the outer circumference and fixed at the inner circumference, as illustrated in *Figure 1*. Zero-velocity boundary conditions normal to all the domain faces prevent the flow of material into or out of the domain.

Results and discussion

The applied boundary conditions fully confine the water inside the Eulerian domain. Because the $U_s - U_p$ material is nearly incompressible, care must be taken to ensure that the applied boundary conditions do not result in a volume change, which could induce spurious pressure oscillations in the water. Indeed, the prescribed tangential velocity is volume-preserving (tangential to the domain boundary) only in infinitesimal deformation. Finite displacement of the boundary nodes occurs in a straight line trajectory, not in a circumferential arc, which induces radial expansion as well as circumferential motion. The underlying Lagrange-plus-remap formulation of the EC3D8R element utilizes an intermediate “deformed” state, which includes this finite displacement (see *Eulerian*

Analysis Techniques). This formulation results in large pressure oscillations that overshadow the viscous shear stresses of interest. The pressure is relieved by releasing the radial velocity at the inner radius, allowing infinitesimal radial motion to offset the radial components of prescribed velocity.

The analytical solution for the steady-state tangential velocity along the radius is given by *Granger (1995)* as

$$V_\theta(r) = \frac{Ub}{a^2-b^2}\left(\frac{a^2}{r} - r\right).$$

In Abaqus/Explicit the simulation begins with the water at rest, and the tangential velocity U is prescribed on the outer boundary. The velocity propagates radially inward via the viscosity and eventually reaches nearly steady-state conditions. With U = 0.2932 m/s, the water disk rotates one revolution in about 0.15 seconds. A simulation time of 3 seconds is chosen so that the water disk rotates 20 revolutions and the solution approximates the steady state. *Figure 2* shows the solutions of the transient tangent velocities along the radius. After 20 revolutions the transient solution closely matches the analytic steady solution.

Without the pressure relief boundary modification, large pressures develop. *Figure 3* shows the evolution of the pressure along the radius. The pressure experiences large oscillations at the beginning of the revolution of t = 0.015 seconds and rapidly increases with time. Indeed, the oscillation is also observed in the velocity curve close to the inner surface at t = 0.015 and 0.15 seconds. A positive tangent velocity here indicates that the water even flows in the reverse direction to the applied velocity close to the inner surface.

Figure 4 and *Figure 5* show the evolution of the tangential velocity and pressure distributions along the radius with the boundary condition at the inner radius relieved so that the boundary nodes can move radially. The tangential velocity gradually approaches the analytical solution. The pressure is reduced by over three orders of magnitudes and oscillates about the analytical solution of a constant pressure of zero. The relieved boundary condition also speeds up the calculations by a factor of nearly 2.5 times. The relieved boundary at the inner radius results in a radial displacement of 4.32×10^{-6} m, or 0.01% of the inner radius the model, which can safely be ignored.

Considering the transient dynamic nature of Abaqus/Explicit, the tangential velocity profile after three seconds shows good accuracy compared to the steady-state analytical solution.

Input files

eulerian_rotating_disk.inp	Rotating water disk.

References

- Granger, R.A., Fluid Mechanics, Dover Publications, 1995.

Tables

Table 1: Material parameters for water.

Parameter	Value
Density (ρ)	998.2 kg/m^3
Viscosity (η)	0.1 N$^\bullet$ s/m^2
c_0	1450 m/s
s	0
Γ_0	0

Figures

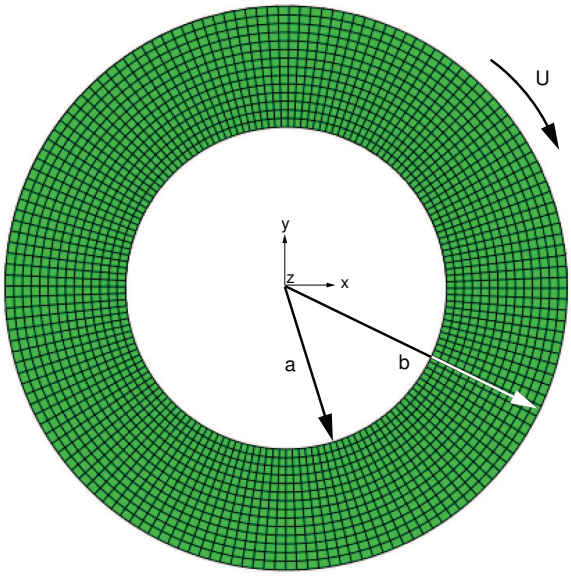


Figure 1: Geometry, mesh, and boundary conditions of the Eulerian domain.

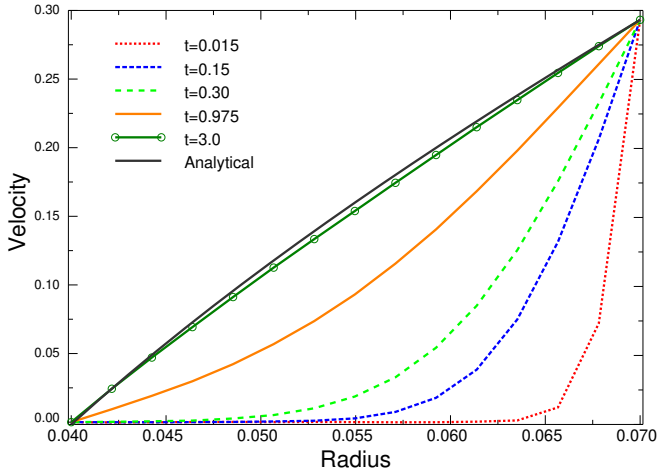


Figure 2: Evolution of the tangential velocities without pressure relief.

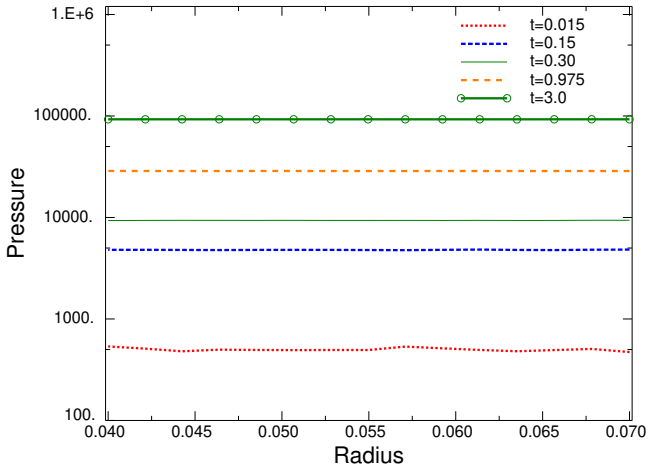


Figure 3: Evolution of the pressure without pressure relief.

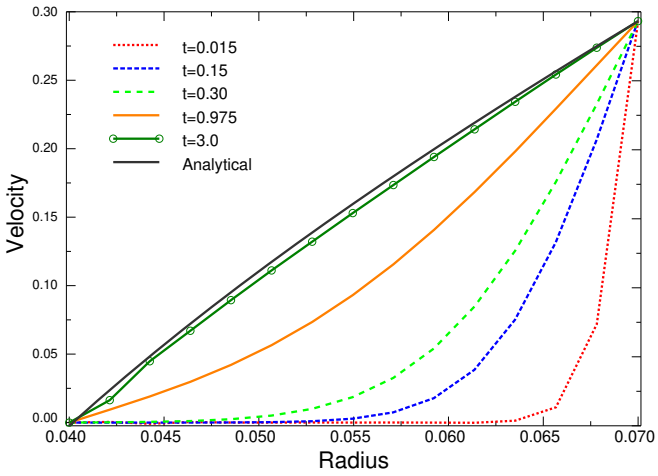


Figure 4: Evolution of the tangential velocities with relieved boundary condition at the inner radius.

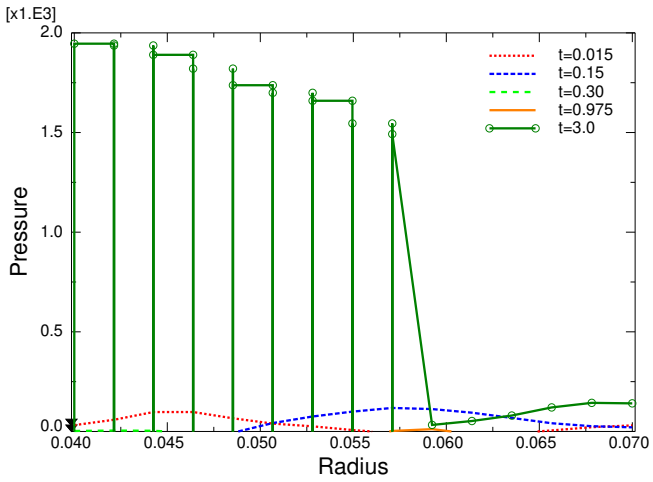


Figure 5: Evolution of the pressure with relieved boundary condition at the inner radius.

Smoothed particle hydrodynamic analysis

Products: Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Bird strike on an airplane engine blade

Elements tested

PC3D

Problem description

This verification problem tests the ability of PC3D elements to describe the impact of a bird on an airplane engine blade. The rotating airplane engine blade is subjected to an impact with a cylindrical model of a flying bird using the smoothed particle hydrodynamic (SPH) technique. After the impact, the bird completely disintegrates and splashes over the surface of the engine blade. A similar approach can be used for modeling severe deformations of thin shell structures impacted by objects moving with high velocity.

Model: This model analyzes the impact interaction between a flying object and a rotating airplane engine blade. The airplane engine blade is modeled using 960 S4RS shell elements. A set of nodes closer to the turbine hub are kinematically coupled to a reference node situated at the center of the hub. A constant angular velocity of 5.466×10^2 rad/s is applied at the reference node about the z-axis. The engine blade is modeled with an elastic-plastic material with Young's modulus $E = 210$ GPa, Poisson's ratio $\nu = 0.3$, density $\rho = 4.0 \times 10^3$ kg/m^3, and isotropic hardening. The flying bird is modeled using 4160 PC3D elements. The bird material is modeled using a tabular equation of state (EOS) material with a tensile failure strength of 94 MPa and a density of $\rho = 1.0 \times 10^3$ kg/m^3. The radius of the cross-section of the cylinder modeling the bird is 0.04 m, and the height of the cylinder is 0.076 m. The contact interaction between the surfaces of the bird object and the shell structure is defined through contact inclusions.

The initial configuration of the model is shown in *Figure 1*.

An intermediate deformed configuration of the airplane engine blade and the bird system is shown in *Figure 2*.

Results and discussion

After the impact, the blade undergoes severe deformation. The edges of the thin shell structure close to the impact area become warped. The bird object completely disintegrates and splashes over the surface of the engine blade. This test problem verifies the capability of the SPH technique to model large deformations and failure of fluid-like materials. Contact interaction between the PC3D and the S4RS elements is also verified.

Input files

ver_prc_birdsplash.inp	Abaqus/Explicit input file.

Figures

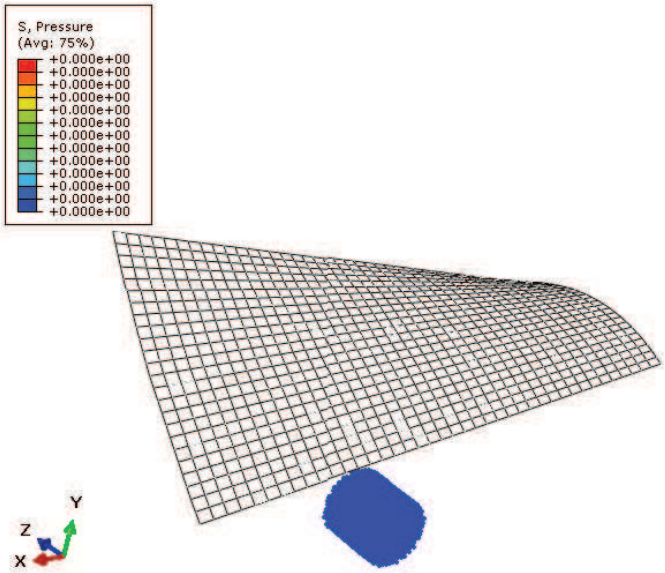


Figure 1: Undeformed configuration of the airplane engine blade and the bird system.

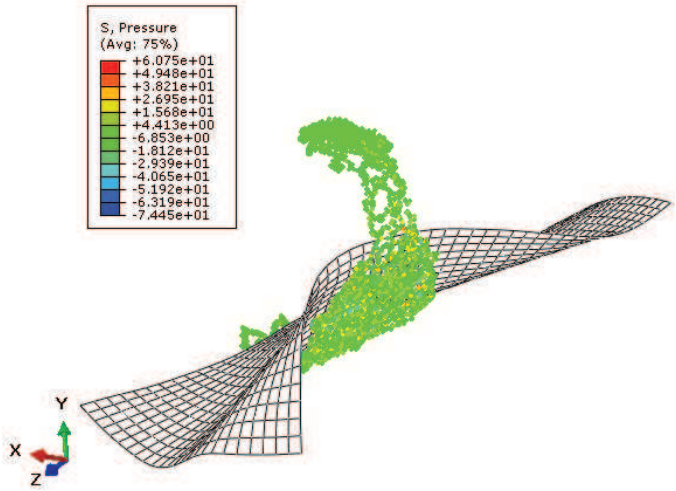


Figure 2: Deformed configuration of the airplane engine blade and the bird system.

Bird strike on an airplane engine blade using conversion

Elements tested

C3D8R PC3D

Problem description

This verification problem tests the ability of reduced-integration continuum elements (C3D8R) elements to convert to SPH particles as deformation progresses during the impact of a bird on an airplane engine blade.

Model: Overall the model, the material properties, and the loading conditions are the same as in *Bird strike on an airplane engine blade*. The only exception is that the bird is first modeled with C3D8R elements rather

than with PC3D elements. A strain-based criterion is used to convert each continuum element to eight SPH particles. The contact interaction between the internally generated particles and the shell structure is defined automatically from the user-defined contact inclusions.

The initial configuration of the model is shown in *Figure 3*.

An intermediate deformed configuration of the airplane engine blade and the bird system is shown in *Figure 4*.

Results and discussion

After the impact, the blade undergoes severe deformation. Continuum elements convert progressively as the specified maximum principal strain is reached in each element. The edges of the thin shell structure near the impact area become warped.

Input files

ver_prc_birdsplashconv.inp	Abaqus/Explicit input file.

Figures

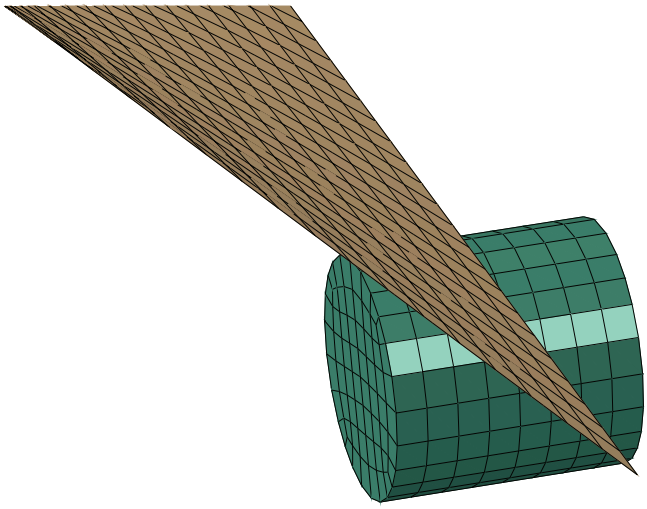


Figure 3: Undeformed configuration of the airplane engine blade and the bird modeled with continuum elements.

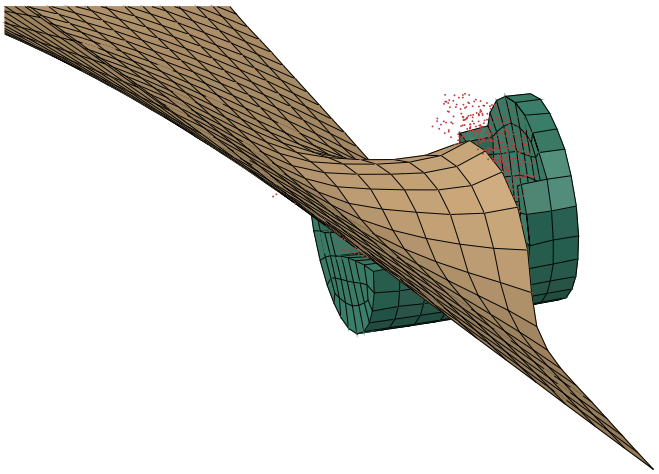


Figure 4: Deformed configuration of the airplane engine blade and the bird as the conversion of the continuum elements progresses.

Water splash in a square pan

Elements tested

PC3D

Problem description

This problem tests the ability of PC3D elements to model impact and mixing of two liquid bodies of the same material. A spherical water drop falls into a square container containing water under gravitational forces. The water drop moves down toward the water in the container and, after splashing, settles to an equilibrium state within the container. The container is modeled using five shell elements. In this test problem mass scaling and bulk modulus reduction are used to increase the value of the stable time increment. Since compressibility does not play a significant role in this analysis, this modeling choice should not affect the results significantly.

Model: This model analyzes the impact and mixing of two liquids with the same material properties. The spherical liquid drop and the liquid in the container are modeled using 3544 and 9000 PC3D elements, respectively. Both liquids are defined using an EOS material of type USUP modeling a linear equation of state. The parameters used in this material model are $c_0 = 0.111803398875$ mm/s, $s = 0.0$, and $\Gamma_0 = 0.0$. To increase the stable time increment, the density of the liquid is artificially defined as $\rho = 1$ tonne/mm^3. The height of the container is 5 mm. The horizontal cross-section of the container has a square shape with a side length of 15 mm. The lateral and bottom walls of the container are modeled as S4R shell elements. The contact interaction between the liquid and the shell structure is defined through contact inclusions.

The initial configuration and an intermediate configuration are shown in *Figure 5* and *Figure 6*.

Results and discussion

This test problem verifies the capability of the SPH technique in Abaqus/Explicit to model the impact and mixing processes of two liquid materials. The mass scaling technique, which drastically increases the stable time increment in this dynamic analysis, is also verified.

Input files

ver_prc_watersplashinpan.inp	Abaqus/Explicit input file.

ver_prc_watersplashinpan_sphere.inp	Sphere nodal coordinates.
ver_prc_watersplashinpan_tank.inp	Tank nodal coordinates.

Figures

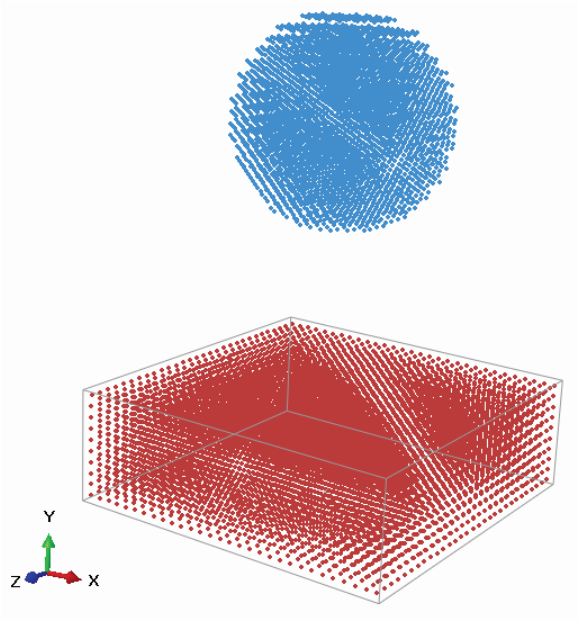


Figure 5: The initial configuration of a water drop and a water-filled square pan.

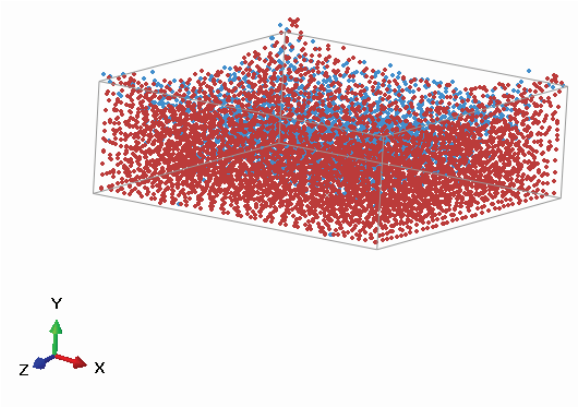


Figure 6: An intermediate configuration of the water drop splashing in a water-filled square pan.

Splashing of a figurehead

Problem description

This problem tests the impact interaction between PC3D elements and a rigid solid structure with a complex curved surface. A block of liquid is moved toward a figurehead and splashes over its surface. The cohesion force used in this model helps maintain some tensile strength for the liquid material during splashing.

Model: This model analyzes the impact interaction between a liquid modeled using the SPH technique and a rigid solid structure. The block of liquid is modeled using 53040 PC3D elements. The material model of the liquid used is an EOS material of type USUP modeling a linear equation of state. The material parameters used are $c_0 = 1.5 \times 10^6$ mm/s, $s = 0.0$, and $\Gamma_0 = 0.0$. A failure strength of 2 MPa is defined for this EOS type material. To model the figurehead, 4084 R3D3 rigid elements are used. The initial velocity of the liquid is 3000 mm/s along the *y*-direction toward the figurehead. The confining box has a dimension of 800 mm × 800 mm × 500 mm, and it is modeled using 48 R3D4 rigid elements. The contact interaction between the liquid and the surfaces of the figurehead and the confining box is defined through contact inclusions with the no-friction surface interaction.

The initial and intermediate configurations are shown in *Figure 7* and *Figure 8*.

Results and discussion

This problem verifies the capability of the SPH technique to model the impact interaction between a block of liquid and a rigid body with a complex surface topography. The effects of the cohesion force modeled for the SPH particles are also tested.

Input files

ver_prc_splashfigurehead.inp	Abaqus/Explicit input file.

Figures

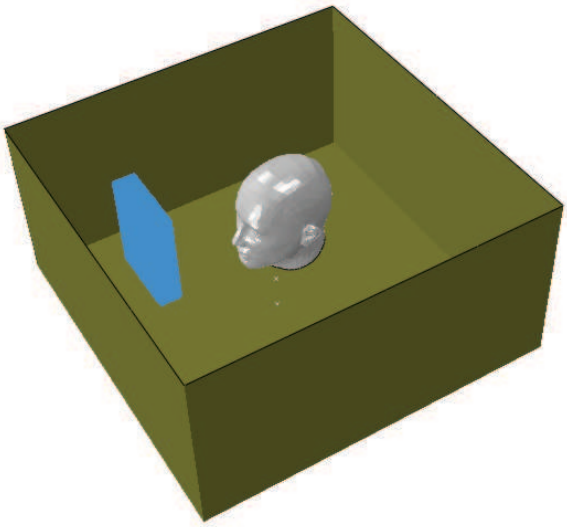


Figure 7: The initial configuration of a block of water and a figurehead.

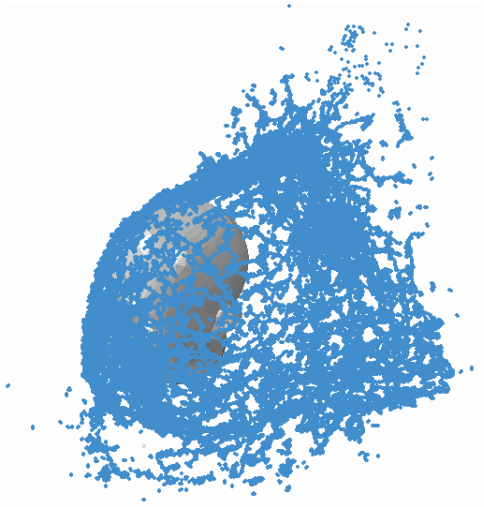


Figure 8: An intermediate configuration of water splashing on a figurehead.

Melting of a figurehead statue

Problem description

This problem tests the ability of PC3D elements to model large deformation and failure of an isotropic elastic-plastic material upon an abrupt change of temperature. A figurehead statue, modeled with temperature-dependent material properties, begins to melt as the temperature suddenly jumps to a higher value. The contact interaction between the SPH related particles and the rigid elements is also tested.

Model: This problem analyzes the temperature-related failure of a figurehead statue modeled using the SPH technique. The figurehead statue is modeled using 8252 PC3D elements, and it is characterized by a temperature-dependent elastic-plastic material mode via field variable dependencies. The Young's modulus, E, is equal to 2 MPa when the nondimensional field variable is equal to 1.0, and it is equal to 0.8 MPa when this variable changes to 2.0. The dependence of the plastic properties on the temperature is given via tabular data.

The density of the material is defined as $\rho = 1.0 \times 10^{-8}$ tonne/mm^3. Fifty R3D4 elements are used to model the bottom and the lateral walls. The melting process of the figurehead statue is accelerated after a sudden rise of the temperature during the dynamic analysis. The contact interaction between the solid statue and the rigid wall is defined through contact inclusions.

The initial and intermediate configurations are shown in *Figure 9* and *Figure 10*.

Results and discussion

This problem verifies the application of the SPH technique to model the large deformation and failure of a temperature-dependent isotropic elastic-plastic material. The contact interaction between this material modeled using SPH particles and the rigid bodies is also tested.

Input files

ver_prc_figureheadmelting.inp	Abaqus/Explicit input file.

Figures

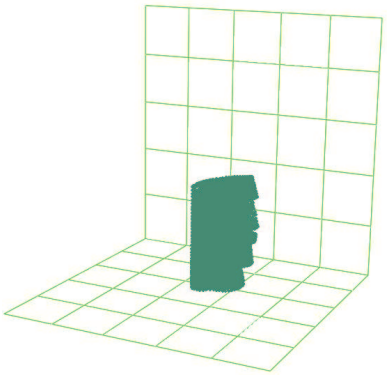


Figure 9: The initial configuration of the figurehead statue.

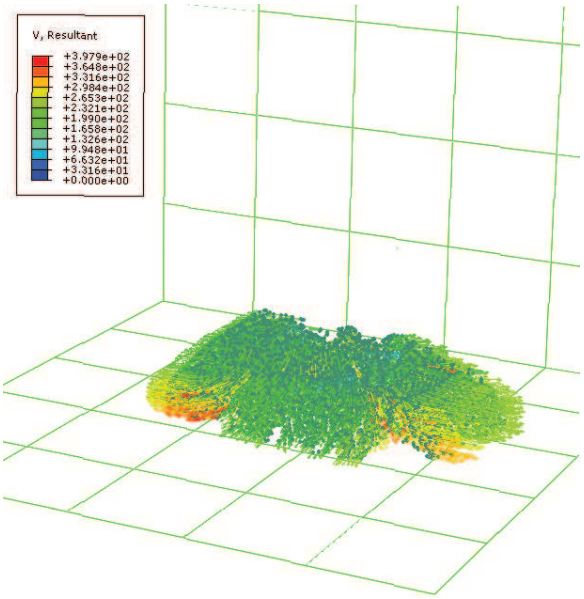


Figure 10: Velocity vector plot of an intermediate configuration for the melted figurehead statue.

Smashing of a figurehead

Problem description

This problem tests the ability of PC3D elements to model the impact of a figurehead on solid walls. The figurehead, modeled as a toothpaste-like viscous material, is smashed onto solid walls. After the impact, the figurehead is completely crashed on the lateral wall and then flows down onto the bottom wall under gravitational forces. The contact interaction between the SPH related particles and the rigid elements is also tested.

Model: This model analyzes the impact interaction between a figurehead and solid walls. The figurehead is modeled using 8252 PC3D elements. The material model used for this figurehead is an EOS material of type USUP modeling a linear equation of state. The material parameters used are $c_0 = 1.405 \times 10^5$, $s = 0.0$, and $\Gamma_0 = 0.0$. A linear viscous shear behavior is defined for this hydrodynamic material through tabular data. A

tensile failure strength of 10 MPa is also defined for this material. The density of the figurehead is set to $\rho = 9 \times 10^{-10}$ tonne/mm^3. Fifty R3D4 elements are used to model the bottom and the lateral walls. The initial velocity of the figurehead is set to 1.0×10^3 mm/s toward the lateral wall. The figurehead then follows a parabolic path under gravitational forces until it strikes the wall. After the impact, the figurehead is smashed onto the lateral wall and then crashes into the corner edge because of complete material failure. The contact interaction between the figurehead and the rigid wall is defined through contact inclusions using rough friction to describe the frictional interactions.

The initial configuration and an intermediate configuration of the figurehead and the rigid walls are shown in *Figure 11* and *Figure 12*.

Results and discussion

This problem verifies the application of the SPH technique to model the failure of a cohesive linear viscous material. The contact interaction between this material and a rigid wall is also tested.

Input files

ver_prc_figureheadsmashing.inp	Abaqus/Explicit input file.

Figures

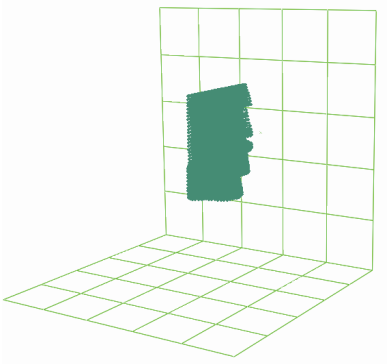


Figure 11: The initial configuration of a figurehead.

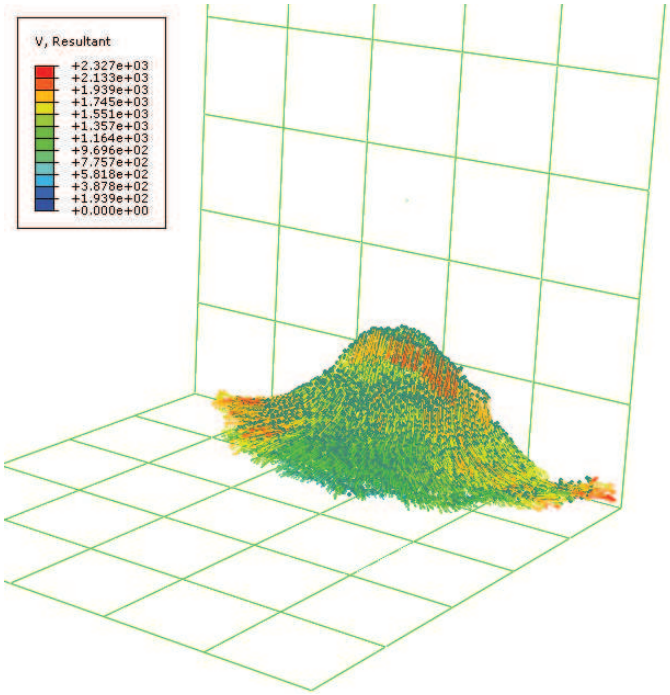


Figure 12: Velocity vector plot of an intermediate configuration for the figurehead after being smashed on a wall.

Projectile impact on a plate

Elements tested

PC3D

Problem description

This verification problem tests the ability of PC3D elements to handle large deformations and failure of a rate-dependent elastic-plastic material upon impact of a high-speed projectile. A solid plate, of which the central part is modeled using the SPH technique, is subjected to an impact by a high-velocity cylindrical rigid object. After the impact, the part close to the center of the plate first undergoes a large deformation and then breaks apart. Eventually, the projectile perforates the plate.

Model: This model analyzes the impact interaction between a high-velocity projectile and a solid plate. The solid plate has a dimension of 400 mm × 400 mm × 12 mm. A circular part with a radius of 100 mm in the center of the plate is modeled using 102726 PC3D elements, and the remaining part of the plate is modeled using 9312 C3D8R elements. The length and radius of the cylindrical rigid solid projectile are 25 mm and 8.4 mm, respectively. The initial speed of the projectile is set to 1000 m/s. The material used for the plate is a steel with

Young's modulus $E = 2.1 \times 10^5$ MPa, Poisson's ratio $\nu =$ 0.3, and density $\rho = 7.8 \times 10^{-9}$ tonne/mm^3. The plate is modeled as an elastic-plastic material with rate-dependent hardening. Ductile and shear damage are evolved based on an energy criterion. The interaction between the rigid projectile and the solid plate is defined using frictional contact with a friction coefficient of 0.3.

The initial configuration of the model is shown in *Figure 13*, and an intermediate deformed configuration cross-section is shown in *Figure 14*.

Results and discussion

After the impact, the center part of the plate first undergoes a large deformation and then breaks into pieces. In the end, the projectile penetrates the plate completely. This problem verifies the ability of the SPH technique to model large deformation and failure of rate-dependent elastic-plastic materials. Contact interaction between the PC3D elements and solid elements is also verified.

Input files

ver_prc_projectileimpact.inp Abaqus/Explicit input file.

Figures

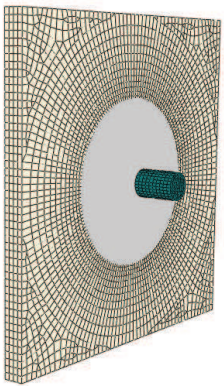


Figure 13: The initial configuration of the solid plate and projectile.

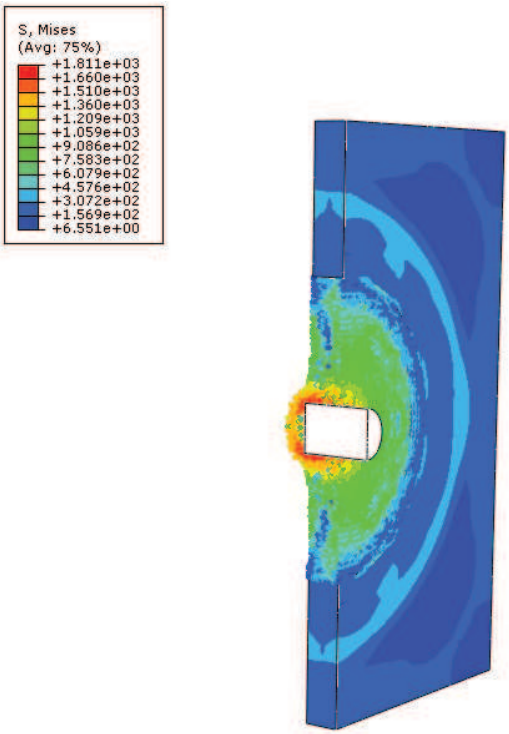


Figure 14: Contour plot of Mises stress for the solid plate subjected to an impact of a projectile.

Stretching test of a solid body using SPH element conversion

Elements tested

C3D8R PC3D

Problem description

This problem verifies the accuracy of Lagrangian element (C3D8R) to SPH particle (PC3D) conversion for a solid body subjected to uniaxial stretching. A cubic shaped solid body, initially modeled with C3D8R elements, is stretched using displacement boundary conditions in the z-direction. One section of elements in the center part is converted to PC3D elements at a certain time (0.1s). The results are compared with the reference solution of a Lagrangian model without element conversion.

Model: The cubic-shaped solid body has dimensions of 1m × 1m × 1m and is modeled with 1331 C3D8R elements. One column of 121 C3D8R elements in the center of the part is converted to 968 PC3D elements at time 0.1s using a user subroutine. The solid cube is subjected to a stretching boundary condition in the z-direction.

The solid cube is modeled using an elastic material with a Young's modulus $E = 2.1 \times 10^5$ Pa, Poisson's ratio $\nu =$ 0.3, and density $\rho = 1000$ kg/m^3. For comparison purposes, the results of an equivalent Lagrangian model with the same dimensions, material properties, and boundary conditions are also presented.

Results and discussion

Figure 15 shows the stress distributions of the two models (the Lagrangian model is on the left, and the SPH element conversion model is on the right). Overall, the stress distributions between the two models are similar. Localized differences exist near the interface between the solid elements and SPH particles for the SPH element conversion case, especially close to the edges of the cube. These differences are due to the differences in the element formulations between the Lagrangian finite element method and the SPH method. As shown in *Figure 16*, the reaction forces of the two models differ by only 1.4% (dashed line: Lagrangian model, solid line: SPH element conversion model), which indicates that forces are transferred correctly between the Lagrangian solid elements and the converted SPH particles. This verifies the accuracy of the SPH element conversion method.

Input files

cube_progressiveconv.inp	Abaqus/Explicit input file for stretching test with SPH element conversion.
vusdfld_conv.f	User subroutine for stretching test with SPH element conversion.
cube_noprogressiveconv.inp	Abaqus/Explicit input file for stretching test with Lagrangian elements.

Figures

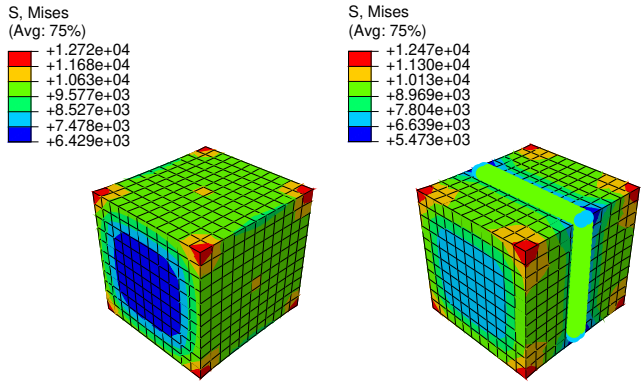


Figure 15: Stress distribution of uniaxial stretching test (Left: Lagrangian model, Right: SPH element conversion model).

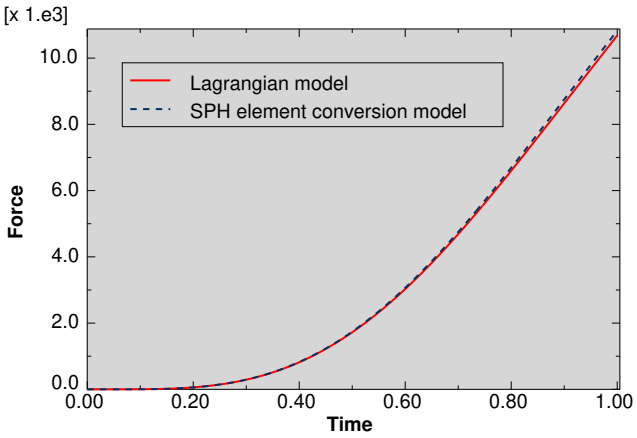


Figure 16: Reaction forces of uniaxial stretching tests (Dashed line: Lagrangian model, Solid line: SPH element conversion model).

Frequency extraction using the AMS eigensolver

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

The tests in this section verify the frequency extraction procedure using the AMS eigensolver in Abaqus/Standard by comparing the results with those obtained by the Lanczos eigensolver.

One-element tests

Elements tested

CPE4 C3D8

Features tested

Eigenvalue extraction for a system with a symmetric stiffness matrix and multi-point constraints, selective modal recovery, full modal recover, and import.

Problem description

The two-dimensional model consists of a linear element of unit length. The nodes at one end ($y = 0$) are constrained, while the nodes at the other end are involved in a LINKMPC. The eigenvalue extraction is performed for the undeformed configuration. The three-dimensional model consists of a single linear element and is mainly used for testing the import feature.

Results and discussion

The eigenvalues obtained for both the AMS and Lanczos procedures are identical.

Input files

ams_1cpe4.inp	Eigenvalue extraction for a model with one element using the AMS eigensolver.
ams_import0.inp	Preloading of a single C3D8 element.
ams_import.inp	Frequency extraction of the import model using the AMS eigensolver.

Model with various lagrange-multiplier constraints (contact, connectors, distributing couplings)

Elements tested

C3D8I C3D8R C3D10M

Features tested

Constraints with Lagrange multipliers and submodeling, mode-based steady-state dynamic restart, and selective modal recovery.

Problem description

The model consists of a semisphere pressed against a cube that is in contact with a rigid surface. The semisphere is also connected to the cube via four axial connectors.

In the preloading step the semisphere is pressed against the cube to establish contact. The load is applied at the reference node of the distributing coupling. In the second step the frequencies of the preloaded structure are extracted via the AMS procedure. Finally, the mode-based steady-state response is calculated in the third step using the results of the frequency extraction step. The results are compared with those obtained by the Lanczos eigensolver.

Results and discussion

In the following table the frequency extraction step results obtained by the Lanczos and AMS eigensolvers are compared.

Mode	Lanczos	AMS
1	11.547	11.551
2	11.916	11.921
3	20.664	20.690
4	25.792	25.840
5	27.916	27.963
6	28.807	28.862
7	42.048	42.110
8	42.370	42.441

Input files

ams_conn_contact.inp	Full analysis using the AMS eigensolver.
ams_conn_contact_res.inp	Mode-based steady-state dynamic analysis restarted from the end of the frequency step.
ams_conn_contact_submodel.inp	Frequency extraction and mode-based steady-state dynamic analysis of a submodel driven entirely from the original model.

Model with coupled-temperature displacement

Elements tested

S8RT B31H B33H B31 B33

Features tested

Coupled temperature-displacement steps, hybrid Bernoulli and Timoshenko beams, full modal recovery, and mode-based steady-state dynamic analysis.

Problem description

The model consists of two rectangular parallel plates connected via beams at each corner. The structure is preloaded by applying a heat flux at the center of the top plate. The linear response is analyzed in a mode-based steady-state dynamic step preceded by a frequency extraction step using the AMS solver.

Results and discussion

In the following table the frequency extraction step results obtained by the Lanczos and AMS eigensolvers are compared.

Mode	Lanczos	AMS
1	14.743	14.745
2	14.743	14.748
3	15.296	15.301
4	17.158	17.164
5	29.476	29.505
6	38.684	38.749
7	38.684	38.773
8	52.778	53.009
9	63.201	63.545
10	67.253	67.621
11	67.253	67.641
12	70.055	70.555
13	87.080	87.166
14	88.789	89.594
15	88.789	89.720
16	88.818	90.292
17	91.946	92.735
18	92.877	93.825

Input files

ams_temp_plates.inp Full analysis using the AMS eigensolver.

Tire model with symmetric model generation and symmetric results transfer

Elements tested

CGAX3H CGAX4H SFMGAX1

Features tested

Eigenvalue extraction for a tire model with hybrid and/or cylindrical elements, axisymmetric model followed by symmetric model generation with symmetric results transfer, and full modal recovery.

Problem description

The axisymmetric tire is inflated and then transferred to a full three-dimensional configuration. Subsequently, the rigid surface is brought in contact with the full tire, obtaining the footprint. Finally, the linear response is analyzed by performing a frequency extraction using the AMS eigensolver followed by a mode-based steady-state dynamic step.

Results and discussion

The following table shows the comparison of eigenfrequencies obtained by the Lanczos and AMS eigensolvers.

Mode	Lanczos	AMS
1	47.552	47.590
2	48.992	49.042
3	54.391	54.445
4	56.749	56.795
5	77.582	77.743
6	82.153	82.265
7	85.123	85.268
8	85.553	85.694
9	98.554	98.802
10	103.73	104.06
11	112.37	112.77
12	116.90	117.47
13	118.64	119.08
14	119.71	120.04
15	124.68	125.18
16	130.75	131.43
17	132.16	132.60
18	136.05	136.61
19	137.41	138.03
20	138.30	139.02
21	140.35	140.97
22	140.58	141.23
23	143.88	144.66
24	144.98	145.75
25	148.05	148.99
26	152.60	153.74

Input files

ams_tire_axisymm.inp	Axisymmetric tire model.
ams_tire_full3d.inp	Three-dimensional tire model.

Model with map solution

Elements tested

CPS3

Features tested

Solution mapping and selective modal recovery.

Problem description

The first model is subject to a static preload. The solution is mapped onto a second mode with different elements, and the structure is further loaded statically. Finally, the eigenvalues of the loaded structure are extracted via the AMS eigensolver.

Results and discussion

The following table shows the comparison of eigenfrequencies obtained by the Lanczos and AMS eigensolvers.

Mode	Lanczos	AMS
1	14.925	14.926
2	43.614	43.617
3	48.566	48.571
4	95.490	95.540

Input files

ams_mapsolution_1.inp	Original model preloaded statically.
ams_mapsolution_2.inp	Solution-mapped model with further preloading and frequency extraction using the AMS eigensolver.

Models with material orientations, nodal transformations, and initial conditions

Elements tested

C3D8 SFM3D4R S4 S8R

Features tested

Material orientations, nodal transformations, initial conditions, selective modal recovery, and full modal recovery.

Problem description

Relatively small problems with simple topologies constructed for testing the features mentioned above.

Results and discussion

The eigenfrequencies obtained by the AMS and Lanczos eigensolvers are identical for the model with material orientations and initial conditions. The model with nodal transformations exhibits differences smaller than 1%.

Input files

ams_material_ori.inp	Model with material orientations and initial conditions.
ams_nodal_transf.inp	Model with nodal transformations.

Models with residual modes

Elements tested

CPE4R C3D20R

Features tested

Residual modes, selective modal recovery, and full modal recovery.

Problem description

Models of simple topology to test the accuracy of residual modes using the AMS eigensolver.

Results and discussion

The following table compares the eigenmodes obtained using the Lanczos and AMS eigensolvers.

Residual	Mode	Lanczos	AMS
no	1	4992.3	4993.1
no	2	5430.0	5430.9
no	3	7340.8	7344.6
no	4	10875.	10877.
no	5	13716.	13724.
yes	6	25445.	24359.
yes	7	36445.	34198.
yes	8	40925.	35377.

The maximum displacement in the steady-state dynamic step at 13kHz is 1.949 units with the Lanczos procedure, versus 1.848 units with the AMS eigensolver.

Input files

ams_resmod_c3d20r.inp	Three-dimensional model with residual modes, AMS, and full modal recovery.
lanczos_resmod_c3d20r.inp	Three-dimensional model with residual modes and Lanczos.
ams_resmod_cpe4r.inp	Two-dimensional model with residual modes, AMS, and selective modal recovery.

Miscellaneous models

Elements tested

SAXA12 M3D4

Features tested

Motion of material through the mesh and section distributions.

Problem description

Models with simple topology to test the features mentioned above.

Results and discussion

The results are identical using both the Lanczos and AMS eigensolvers for the model with material motion. For the model with section distributions and SAXA12 elements the results differ slightly in the fifth eigenvalue, as shown in the table below.

Mode	Lanczos	AMS
1	531.33	531.33
2	771.31	771.31
3	1017.6	1017.6
4	1129.4	1129.4
5	1217.0	1217.1
6	1639.2	1639.2
7	1754.1	1754.1
8	2275.6	2275.6
9	3382.0	3382.0
10	3490.5	3490.5
11	3556.7	3556.7
12	3994.9	3994.9

Input files

ams_motion.inp	Model with material motion.
lanczos_resmod_c3d20r.inp	Model with section distributions and SAXA12 elements.

Steady-state dynamics with nondiagonal damping using the AMS eigensolver

Products: Abaqus/Standard Abaqus/AMS

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

The tests in this section verify the mode-based steady-state dynamic analysis procedure supporting nondiagonal damping (structural, viscous, material, and global damping) using the AMS eigensolver in Abaqus/Standard. As a reference solution, the results obtained by the subspace-based steady-state dynamic analysis procedure using the Lanczos eigensolver are used. Some tests are compared to the steady-state direct method.

One-element tests

Elements tested

CPE4 C3D8

Features tested

Mode-based steady-state dynamic step using the eigensolution computed by the AMS eigensolver for a system with material damping, global damping, and the damping controls option.

Problem description

The two-dimensional model consists of a linear element of unit length with material damping. The nodes at the bottom ($y = 0.0$) are constrained, and real and imaginary parts of the concentrated loads are applied to the nodes at the top ($y = 1.0$) . The three-dimensional model is used for testing the selecting eigenmodes and selective modal recovery features.

Results and discussion

The nodal variables at the requested frequency obtained by both the mode-based steady-state dynamic analysis procedure using the AMS eigensolver and the subspace-based steady-state dynamic analysis procedure using the Lanczos eigensolver are identical.

Input files

ssd_ams_1cpe4.inp	Mode-based steady-state dynamic analysis using the AMS eigensolver (CPE4).
ssd_lnz_1cpe4.inp	Subspace-based steady-state dynamic analysis using the Lanczos eigensolver (CPE4).
ssd_lnz_1cpe4_sdamp.inp	Mode-based steady-state dynamic analysis using the Lanczos eigensolver (CPE4) including global damping and damping controls.
ssd_ams_1c3d8.inp	Mode-based steady-state dynamic analysis using the AMS eigensolver (C3D8). Global damping and damping controls tested.
ssd_lnz_1c3d8.inp	Subspace-based steady-state dynamic analysis using the Lanczos eigensolver (C3D8).

Model with discrete material damping

Elements tested

CONN3D2 SPRING1 DASHPOT1 MASS T3D2

Features tested

Mode-based steady-state dynamic step using the eigensolution computed by the AMS eigensolver for a system with discrete material damping (connector damping and dashpot). Global damping and damping controls options are tested here.

Problem description

The simple one degree of freedom model consists of three components: a spring, a mass, and a dashpot. Left-hand sides of the spring and the dashpot are connected to the ground, and the mass element is attached to the right-hand sides of the spring and the dashpot. A unit concentrated load is applied to the mass element in the direction of degree of freedom 1.

The connector model consists of three Cartesian-type connectors that are sequentially connected together. It has two degrees of freedom, and complex connector loads are applied on the two middle nodes.

Results and discussion

The results from the mode-based steady-state dynamic analysis procedure using the AMS eigensolver and the subspace-based steady-state dynamic analysis procedure using the Lanczos eigensolver for the spring-mass-damper system are identical in the frequency range of interest.

For the connector model, the results from the mode-based steady-state dynamic analysis procedure using the AMS eigensolver and the subspace-based steady-state dynamic analysis procedure using the Lanczos eigensolver are identical in the frequency range of interest.

Input files

ssd_ams_1dof.inp	Mode-based steady-state dynamic analysis using the AMS eigensolver for a spring-mass-dashpot model with one degree of freedom.
ssd_lnz_1dof.inp	Subspace-based steady-state dynamic analysis using the Lanczos eigensolver for a spring-mass-dashpot model with one degree of freedom.
ssd_ams_conn3d.inp	Mode-based steady-state dynamic analysis using the AMS eigensolver for a three-dimensional connector element model with connector damping. Global damping and damping controls tested.
ssd_lnz_conn3d.inp	Subspace-based steady-state dynamic analysis using the Lanczos eigensolver for a three-dimensional connector element model with connector damping. Global damping and damping controls tested.
t3d2_ssd_ams_sdamping.inp	Mode-based steady-state dynamic analysis using the AMS eigensolver tested with global damping and damping controls.

Model with frequency-dependent material

Elements tested

CPS4

Features tested

Mode-based steady-state dynamic step for a system with frequency-dependent viscoelastic material and property evaluation feature in the frequency extraction step.

Problem description

The two-dimensional model is a simple cantilever beam model with 12 CPS4 elements. Left-end nodes of a cantilever beam are fixed, and 1.0 GPa is applied to the top surface of the cantilevered beam. Frequency-domain viscoelastic material is defined in a tabular form.

Results and discussion

The results from the mode-based steady-state dynamic analysis procedure at about every 10 Hz are compared with the results from the subspace-based steady-state dynamic analysis procedure with the Lanczos eigensolver, as shown in the table below.

Frequency	SSD with AMS		SSD, SP with Lanczos	
	Magnitude	Phase	Magnitude	Phase
9.08 Hz	–2.714	1.5522e-03	2.714	180.0
19.18 Hz	5.580	179.9	5.581	179.9
29.29 Hz	7.326	0.2724	7.235	0.2676
39.39 Hz	1.751	8.055e-02	1.751	7.9158e–02
49.49 Hz	0.9103	6.2855e-04	0.9103	3.8876e-02
59.59 Hz	0.5928	8.5927e-03	0.5928	8.4387e-03
69.69 Hz	0.4381	–3.2477e-02	0.4381	–3.1929e-02
79.80 Hz	0.3575	–0.1051	0.3575	–0.1033
91.92 Hz	0.3248	–0.2611	0.3248	–0.2566
100.00 Hz	0.3514	–0.6925	0.3507	–0.6808

Input files

ssd_ams_viscoe_cps4.inp — Mode-based steady-state dynamic analysis using the AMS eigensolver for a two-dimensional model with frequency-domain viscoelasticity

ssd_lnz_viscoe_cps4.inp — Subspace-based steady-state dynamic analysis using the Lanczos eigensolver for a two-dimensional model with frequency-domain viscoelasticity

Model with base motion

Elements tested

B23

Features tested

Mode-based steady-state dynamic step with base motion, eigenmode selection, and beam general section along with material damping.

Problem description

The model consists of 20 Euler-Bernoulli beams sequentially connected; each end of the beams is constrained to the ground. Primary base motion is prescribed with user-defined amplitude, and the first 25 modes are selected for mode-based steady-state dynamic analysis.

Results and discussion

The results from both the mode-based steady-state dynamic analysis procedure and the subspace-based steady-state dynamic analysis procedure for this model are identical.

Input files

ssd_lnz_base_b23.inp	Two-dimensional model for a subspace-based steady-state dynamic analysis with base motion, selective eigenmodes, and Lanczos eigensolver.

SIM-based steady-state analysis with multiple load cases

Elements tested

B21 DASHPOTA

Features tested

SIM-based steady-state dynamic analysis with multiple load case definitions.

Problem description

Model: A cantilever beam with a dashpot at the tip.

Material: Young's modulus = 2.0 × 105, Poisson's ratio = 0.3, density = 2.0×10^{-6}. Dashpot damping is frequency dependent as follows:

Table 1: Dashpot damping.

Frequency (Hz)	Damping value
0.0	0.01
100	0.001
200	0.0005

The beam is fixed at one end and is free at the other. The dashpot is connected to the tip and grounded at the other end. A concentrated load of amplitude 1200 is applied at the tip of the cantilever beam. For the second load case the same load is applied as an imaginary part of the load for comparison. The steady-state dynamic analysis is run from 0 to 100 Hz using subspace projection based on modes computed up to 200 Hz.

Results and discussion

The results from the two load cases match in magnitude to the results from a single load case step. The results from the imaginary load case are off by 90° in phase as expected. The following table shows the peak response values:

Frequency (Hz)	Single load		Real load case		Imaginary load case	
	Magnitude	Phase	Magnitude	Phase	Magnitude	Phase
5.85466	2978.0	90.0	2978.0	90.0	2978.0	−180.0
35.4301	625.0	90.0	625.0	90.0	625.0	180.0
97.0515	474.8	90.0	474.8	90.0	474.8	180.0

Input files

cant_dash_ssds_mlc.inp SIM-based steady-state dynamic analysis of the cantilever beam with dashpot, subspace, and multiple load cases. Units: mm, N, MPa.

Media transport

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Media transport analysis using import from Abaqus/Standard to Abaqus/Explicit

Elements tested

C3D8R
M3D4R

Problem description

The verification test in this section is a media transport analysis of a periodic media consisting of nine blocks modeled using M3D4R and C3D8R elements. The membrane elements are used to model the conveyor belt, and the brick elements are used to model the packages on top of the belt. The packages are tied to the belt with a tie constraint. The model is pre-stretched using Abaqus/Standard and imported to Abaqus/Explicit, where the periodic media is defined and activated. The belt is set in motion at the beginning of the Abaqus/Explicit analysis by specifying a uniform initial velocity. The model is illustrated in *Figure 1*.

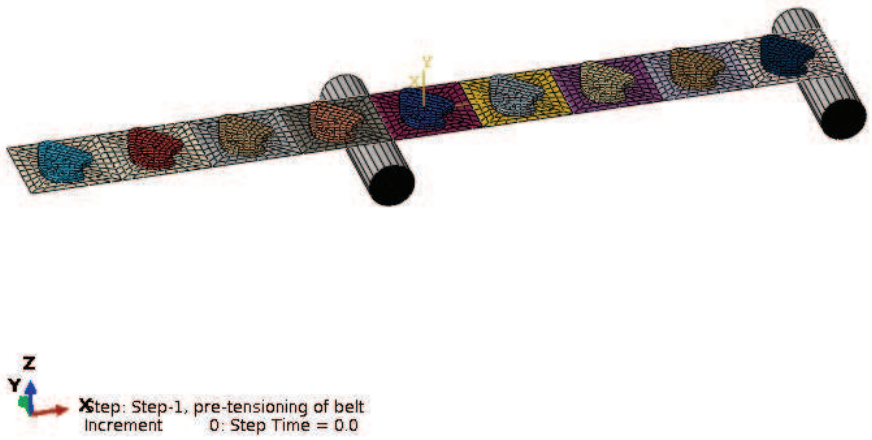


Figure 1: Periodic media transport analysis.

Because the periodic media analysis technique is not available in Abaqus/Standard, the ties between blocks must be defined explicitly using a tie constraint and the boundary conditions at the inlet and outlet must be defined directly at the nodes. In addition, the front end nodes of the inlet block must be constrained to have identical displacements as their corresponding nodes in the back end of the inlet. This is accomplished by defining an equation constraint between corresponding nodes that forces the y-direction displacements to be equal. The belt is stretched by fixing the inlet nodes and displacing the outlet nodes in the x-direction.

In the Abaqus/Explicit analysis the periodic media is defined using element sets and node sets. Two rollers are added with general contact defined between the belt and the rollers. Both the inlet and outlet control nodes are fixed in the y- and z-directions and given a velocity of 1000 in the x-direction. All of the belt and package nodes are given an initial velocity of 1000 in the x-direction. Block shuffling takes place when the back end of the inlet passes the trigger plane. The trigger plane is located at an x-coordinate of –200 and is normal to the x-direction.

Results and discussion

The belt is stretched in the Abaqus/Standard analysis, and the stress state and deformed configuration are imported properly to Abaqus/Explicit. In the Abaqus/Explicit analysis the belt moves through the process zone in a steady-state manner while maintaining the stretched stress state. The stress state remains nearly constant even as blocks are shuffled when the inlet blocks pass through the trigger plane.

Input files

Abaqus/Standard analysis file

belt_standard.inp	First Abaqus/Standard analysis.

Abaqus/Explicit analysis file

belt_explicit.inp	Second Abaqus/Explicit import analysis.

Multistep media transport analysis using Abaqus/Explicit

Elements tested

M3D4R

Problem description

The verification test in this section is the same model described in the previous section, except that the pre-stretching step is defined directly in the Abaqus/Explicit analysis and the packages are fixed to the belt using cohesive contact instead of a tie constraint. The periodic media is inactive during the pre-stretching step. The stretching is achieved by applying velocity boundary conditions with different amplitudes at both inlet and outlet control nodes. The velocity amplitudes ramp up from zero to one in a span of 0.2 seconds; however, one amplitude starts ramping at the start of the step and the other is delayed by 0.2 seconds. At the end of the step the inlet and outlet control nodes have uniform velocity boundary conditions. The periodic media is activated in the second step, allowing block shuffling to take place when the back end of the inlet block passes the trigger plane. The model is illustrated in *Figure 1*.

Results and discussion

The mismatch in inlet and outlet velocities properly generates the stretched stress state in the first step. In the second step the velocities are constant and the belt moves through the process zone in a steady-state manner. The stress state remains nearly constant even as blocks are shuffled when the inlet blocks pass through the trigger plane.

Input files

belt_two_step.inp	Two-step Abaqus/Explicit analysis.

Transferring results between dissimilar meshes in Abaqus/Standard

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D8R C3D10 C3D10M C3D20 C3D4 C3D8T C3D20RT
CAX3H CAX4 CGAX4HT CGAX8RT
CPE3 CPE4 CPE6 CPE6H CPE8 CPE8R
CPEG3HT CPEG4HT
C3D8P C3D20RP CAX4P CAX8RP
CPS3 CPS4 CPS4T DC2D4

Features tested

Solution variables transfer capabilities

Problem description

The verification tests in this section consist of pairs of models. Within each pair the first, or ancestor, model undergoes a simple deformation to a deformed configuration. The second, or descendent, model represents the deformed configuration of the ancestor with a different mesh and possibly with different element types. The solution from the ancestor model is transferred to the descendent model, and the resulting state of this model is verified to be consistent with the ancestor in its deformed configuration.

Model: The ancestor model has a simple rectangular geometry. In most cases the model contains two distinct material regions, shown in *Figure 1*. This model undergoes a uniform compression, as shown in *Figure 2*, and the resulting configuration is chosen as the geometry for the descendent model, as shown in *Figure 3*. Models with axisymmetric elements are placed at a large radial position so that the element behavior is near to that of plane strain elements. Models with three-dimensional elements have a depth of 10 units and have meshes slightly different from those shown in the following planar mesh figures.

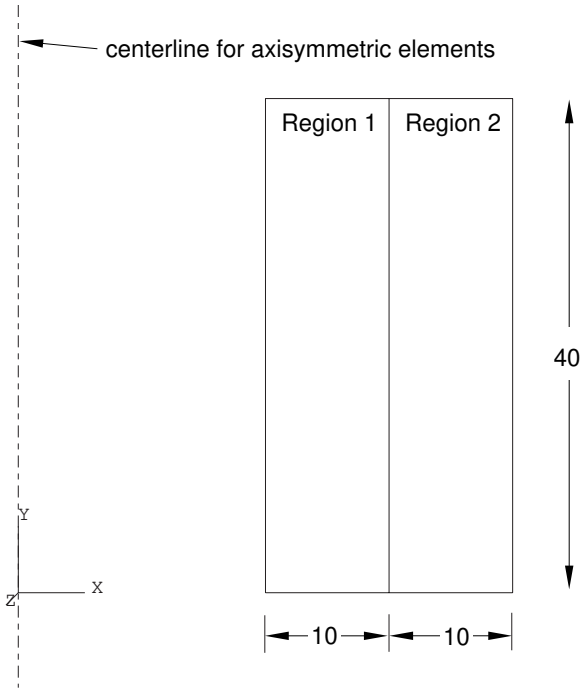


Figure 1: Ancestor model geometry.

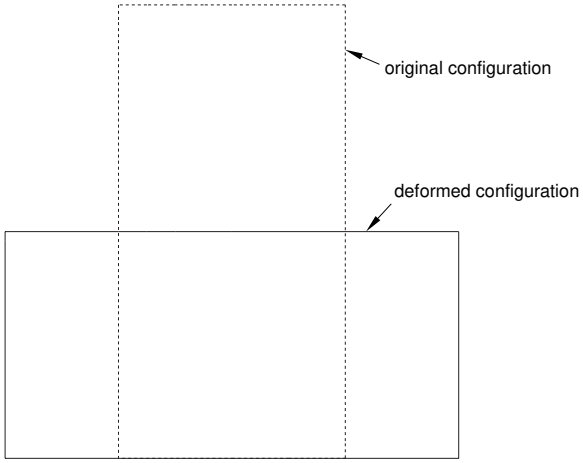


Figure 2: Deformation of the ancestor model.

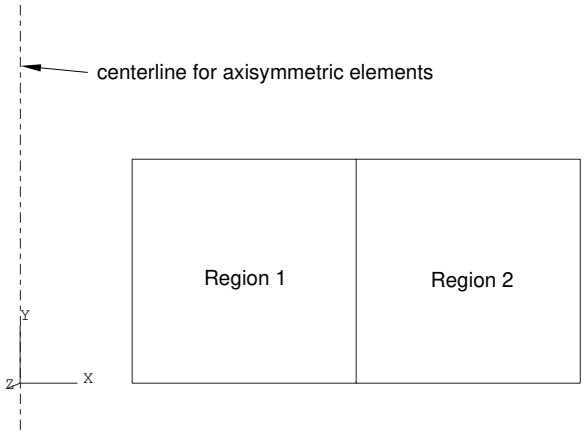


Figure 3: Descendent model geometry.

Mesh: Nonuniform meshes are chosen, as illustrated in *Figure 4*, *Figure 5*, *Figure 6*, and *Figure 7*.

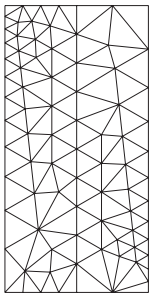


Figure 4: Ancestor model triangular mesh.

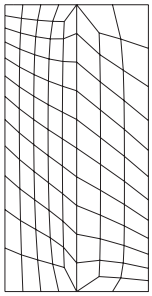


Figure 5: Ancestor model quadrilateral mesh.

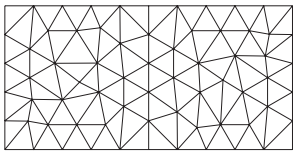


Figure 6: Descendent model triangular mesh.

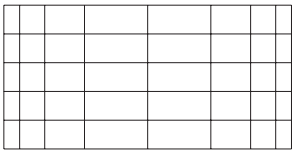


Figure 7: Descendent model quadrilateral mesh.

Material: Material properties are selected from among the following models. In cases where two different material properties are used in adjacent regions, the parameters listed first are applied to one material region and the parameters listed second are applied to the other:

Elastic (including *UMAT* implementation)

Young's modulus	1×10^4 and 1×10^5
Poisson's ratio	0.3

Elastic/plastic

Young's modulus	1×10^4 and 1×10^5
Poisson's ratio	0.3
Yield stress	8×10^3 and 8×10^4

Hyperelastic

C10	1.9×10^3
D11	2.4×10^{-4}

Boundary conditions: The ancestor model is constrained from vertical motion on the bottom surface and from horizontal motion along the interface between the material regions. The top surface is then compressed with a uniform motion while the sides expand with a prescribed, volume preserving motion. These boundary conditions result in a deformed configuration that is independent of the material models used in the analysis. In some tests the deformed configuration shown in *Figure 2* is reached at an intermediate step and increment, which enables testing of solution mapping from intermediate configurations.

Ancestor models with temperature degrees of freedom have a temperature of zero prescribed on the lower boundary and a temperature of 1000 prescribed on the upper boundary, resulting in a linear variation in temperature across the height of the model.

Ancestor models with pore pressure degrees of freedom have a pore pressure of zero prescribed on the lower boundary and a pore pressure of 1 prescribed on the upper boundary, resulting in a linear variation in pore pressure across the height of the model.

Results and discussion

The material solution variables in each descendent model are verified to match those in the ancestor model in its deformed configuration. In cases where the models have distinct material regions, the solution variables in the descendent model are verified to be distinct with no smoothing across the material boundary. The linear distribution in temperature in models with temperature degrees of freedom and in pore pressure in models with pore pressure degrees of freedom is verified to agree between the ancestor and the descendent model.

Input files

The input file names describe the analysis procedure, element type, and material type. The input files are grouped in pairs; each pair is comprised of an ancestor model, from which the solution is transferred, and a descendent model, to which the solution is transferred.

The ancestor analysis files follow the format pmap_*element_material_options*_a.inp; the descendent analysis files follow the format pmap_*element_material_options*_d.inp.

element indicates the element type or types used in the analysis. *material* indicates the type of material in the analysis. *options* indicates the particular procedure or feature tested.

CPE8 element tests:

pmap_cpe8_elastic_static_a.inp	Ancestor model.
pmap_cpe8_elastic_static_d.inp	Descendent model.

CPE4 element tests:

pmap_cpe4_elastic_static_a.inp	Ancestor model.
pmap_cpe4_elastic_static_d.inp	Descendent model.

CPE4 element tests with orientations defined in the ancestor model:

pmap_cpe4_elastic_orient_a.inp	Ancestor model.
pmap_cpe4_elastic_noorient_d.inp	Descendent model.

CPE4 element tests with orientations defined in the descendent model:

pmap_cpe4_elastic_noorient_a.inp	Ancestor model.
pmap_cpe4_elastic_orient_d.inp	Descendent model.

Tests of solution mapping from CPS4 to CPS3 elements:

pmap_cps4_elastic_static_a.inp	Ancestor model.
pmap_cps3_elastic_static_d.inp	Descendent model.

Tests of solution mapping from CPE3 to CPE4 elements:

pmap_cpe3_plastic_static_a.inp	Ancestor model.
pmap_cpe4_plastic_static_d.inp	Descendent model.

Tests of solution mapping from CPEG3HT to CPEG4HT elements:

pmap_cpeg3ht_plastic_static_a.inp	Ancestor model.
pmap_cpeg4ht_plastic_static_d.inp	Descendent model.

Tests of solution mapping from C3D8P to C3D20RP elements:

pmap_c3d8p_elastic_a.inp	Ancestor model.
pmap_c3d20rp_elastic_d.inp	Descendent model.

Tests of solution mapping from C3D8P to C3D20RP elements in a steady soils procedure:

pmap_c3d8p_elastic_ss_a.inp	Ancestor model.
pmap_c3d20rp_elastic_ss_d.inp	Descendent model.

Tests of solution mapping from C3D8P to C3D10MP elements:

pmap_c3d8p_elastic_a.inp	Ancestor model.

pmap_c3d10mp_elastic_d.inp Descendent model.

Tests of solution mapping from C3D8P to C3D8RP elements:

pmap_cpe8p_elastic_a.inp Ancestor model.
pmap_cpe8rp_elastic_d.inp Descendent model.

Tests of solution mapping from CPE8R to CPE6H elements with a user material definition:

pmap_cpe8r_user_static_a.inp Ancestor model.
pmap_cpe6h_user_static_d.inp Descendent model.

Tests of solution mapping from CGAX4HT to CGAX8RT elements:

pmap_cgax4ht_plastic_coupled_a.inp Ancestor model.
pmap_cgax8rt_plastic_coupled_d.inp Descendent model.

Tests of solution mapping from CAX4P to CAX8RP elements:

pmap_cax4p_plastic_a.inp Ancestor model.
pmap_cax8rp_plastic_d.inp Descendent model.

Tests of solution mapping from CPE6 to CPE8 elements with a hyperelastic material defined:

pmap_cpe6_hyperelastic_static_a.inp Ancestor model.
pmap_cpe8_hyperelastic_static_d.inp Descendent model.

Tests of solution mapping from CPS4T to DC2D4 elements:

pmap_cps4t_plastic_coupled_a.inp Ancestor model.
pmap_dc2d4_plastic_heattransfer_d.inp Descendent model.

Tests of solution mapping from C3D8R to C3D10 elements:

pmap_c3d8r_elastic_static_a.inp Ancestor model.
pmap_c3d10_elastic_static_d.inp Descendent model.

Tests of solution mapping from C3D10M to C3D20 elements:

pmap_c3d10m_plastic_static_a.inp Ancestor model.
pmap_c3d20_plastic_static_d.inp Descendent model.

Tests of solution mapping from C3D4 to C3D10M elements with a rotation applied to the ancestor model:

pmap_c3d4_elastic_rotated_a.inp Ancestor model.
pmap_c3d10m_elastic_rotated_d.inp Descendent model.

Tests of solution mapping from C3D8T to C3D20RT elements:

pmap_c3d8t_elastic_coupled_a.inp Ancestor model.
pmap_c3d20rt_elastic_coupled_d.inp Descendent model.

One-step inverse analysis

Products: Abaqus/Standard

References:

- **ONE-STEP INVERSE*

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

S3R S4 S4R STRI3

Features tested

This section provides basic verification tests of the one-step inverse analysis.

Problem description

The tests provide verification of the one-step inverse analysis described in *One-Step Inverse Analysis*. The analysis determines an initial undeformed configuration of a structure given a final deformed configuration and a blank surface. The initial location of the plate is expected to conform with the blank surface.

Model The model consists of a plate approximated with shell elements and a rigid surface.

Mesh The plate is modeled with a mesh of 10 quadrilateral elements or a mesh of 20 triangular elements.

Material A linear isotropic elastic material is used with:

Young's modulus	$E=195 \times 10^6$
Poisson's ratio	$\upsilon=0.33$

(The units are not important.)

Initial Conditions The initial conditions specify an estimate of the unfolded coordinates of all nodes in the plate, which is used a starting point in the iterative algorithm of the one-step inverse analysis.

Boundary conditions All degrees of freedom of the nodes on one side of the plate are fixed. As a result, these nodes are constrained at the position prescribed with the initial conditions.

Results and discussion

All tests using different element types have the expected results. The calculated initial configuration of the plate conforms to the blank surface. In addition, differences between results obtained using different element types are negligible.

Input files

oneStepInverse_s3r.inp	The model using S3R elements.
oneStepInverse_s4.inp	The model using S4 elements.
oneStepInverse_s4r.inp	The model using S4R elements.
oneStepInverse_stri3.inp	The model using STRI3 elements.

Ramping of initial stress

Products: Abaqus/Explicit

References:

- *Ramping of Initial Stresses in Abaqus/Explicit*

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

B31 C3D8 C3D8R CAX6M CPE4 CPE6M S4R S4RS

Problem description

In Abaqus/Explicit you can define ramping of initial stresses for different types of elements in order to introduce the initial stresses gradually based on an amplitude definition. This method helps to reduce initial oscillations in the solution when the initial stress field is not in perfect equilibrium with the external loads. This capability is tested for different element types and materials.

Results and discussion

Each test consists of two elements of the same type and with the same material definition (either hyperelastic or plastic). The ramping of initial stress is defined for one of the elements, and the stress history results are compared between the two elements. In each case, it is verified that the initial oscillations in the solution are reduced for the element that includes ramping of initial stresses. *Figure 1* shows the results for the case of two C3D8R elements with hyperelastic response.

Input files

rampinistress-c3d8r-hyper.inp	Ramping of initial stress test for C3D8R element with hyperelastic material.
rampinistress-c3d8r-mises.inp	Ramping of initial stress test for C3D8R element with plastic material.
rampinistress-c3d8-hyper.inp	Ramping of initial stress test for C3D8 element with hyperelastic material.
rampinistress-c3d8-mises.inp	Ramping of initial stress test for C3D8 element with plastic material.
rampinistress-cpe4.inp	Ramping of initial stress test for CPE4 element with elastic material.
rampinistress-t6m.inp	Ramping of initial stress test for CPE6M element with elastic material.
rampinistress-cax6m.inp	Ramping of initial stress test for CAX6M element with elastic material.
rampinistress-s4r.inp	Ramping of initial stress test for S4R element with elastic material.

rampinistress-s4rs.inp	Ramping of initial stress test for S4RS element with elastic material.
rampinistress-b31.inp	Ramping of initial stress test for B31 element with elastic material.

Figures

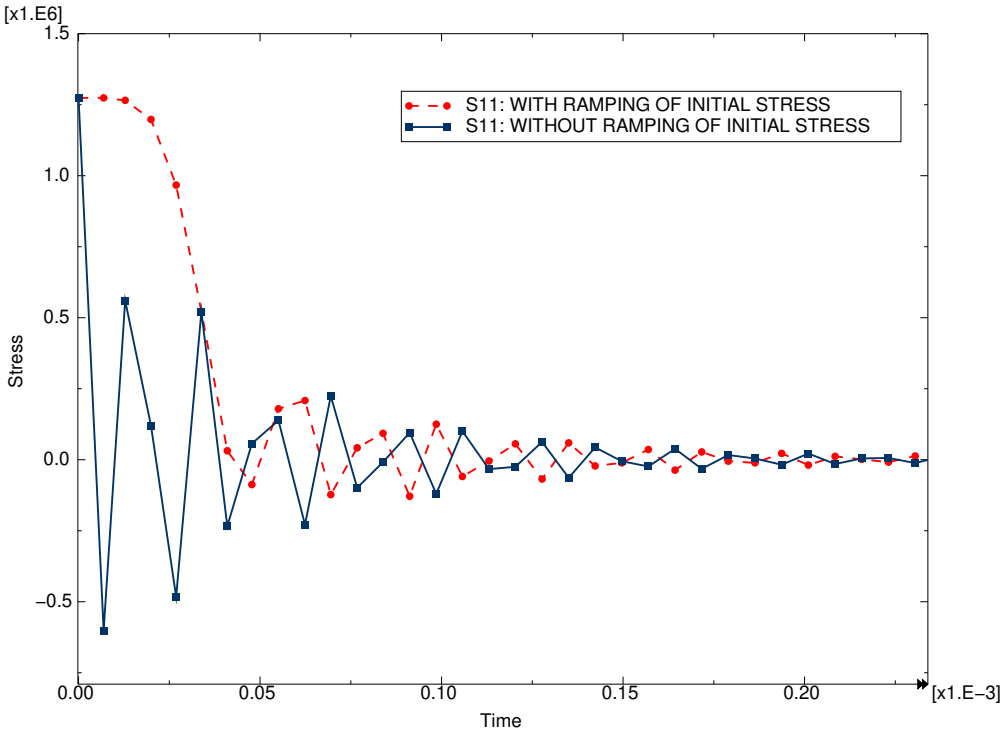


Figure 1: Verification for ramping of initial stress.

Undrained pore fluid flow and stress analysis

Products: Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D4 C3D5 C3D6 C3D8 C3D8R C3D10M
CPE3 CPE4R CPE6M
CAX3 CAX4R CAX6M

Features tested

Undrained pore fluid flow and stress analysis.

Problem description

Abaqus uses a unit cube of geometry with distorted meshes to test three-dimensional solid elements and a simple rectangular geometry to test plane strain elements as well as axisymmetric solid elements. The skeleton materials are linear isotropic elastic with $E = 3000$, $\rho = 1200$, and $\nu = 0.3$. The permeating pore fluid has a density, $\rho_w = 1000$, and a bulk modulus, $K_w = 2.1e^9$. The bulk modulus of the grains is $K_g = 3.7e^7$. The medium is subject to prescribed displacement boundary conditions to control volumetric deformation. You can compare the numerical results with a reference solution.

Reference solution

The reference solution is based on the analytical equations of a fully saturated medium with prescribed volumetric deformation under undrained conditions. This leads to a nonlinear system of equations that is solved numerically.

Results and discussion

All tests using different element types produce the expected results, and the pore pressure agrees well with the reference solution. *Figure 1* shows the pore pressures obtained by the different three-dimensional solid elements, including the element C3D8R with an Arbitrary Lagrangian-Eulerian (ALE) adaptive mesh domain, compared with the reference solution as a function of time. *Figure 2* shows the results of plane strain elements, including CPE4R with an Arbitrary Lagrangian-Eulerian (ALE) adaptive mesh domain, compared with the reference solution.

Input files

c3d4_undrain.inp	C3D4 element.
c3d5_undrain.inp	C3D5 element.
c3d6_undrain.inp	C3D6 element.
c3d8r_undrain.inp	C3D8R element.
c3d8_undrain.inp	C3D8 element.

c3d10m_undrain.inp	C3D10M element.
cpe3_undrain.inp	CPE3 element.
cpe4r_undrain.inp	CPE4R element.
cpe6m_undrain.inp	CPE6M element.
cax3_undrain.inp	CAX3 element.
cax4r_undrain.inp	CAX4R element.
cax6m_undrain.inp	CAX6M element.
ale_c3d8r_undrain.inp	C3D8R element with ALE.
ale_cpe4r_undrain.inp	CPE4R element with ALE.

Figures

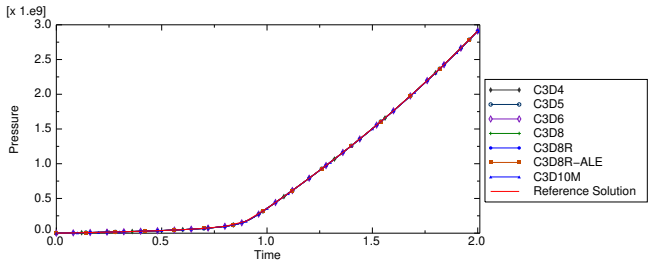


Figure 1: Pore pressures with different three-dimensional elements.

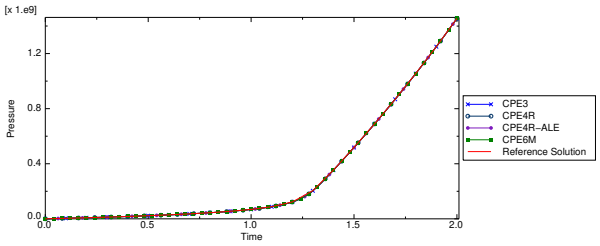


Figure 2: Pore pressures with different plane strain elements.

Controlling material rate dependence

Products: Abaqus/Standard Abaqus/Explicit

References:

- *Rate-Dependent Yield and Friction*
- *Controlling Material Rate Dependence*

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Rate-dependent yield

Elements tested

C3D8 C3D8R

Features tested

Controlling the material rate-dependence of materials with rate-dependent yield.

Problem description

The tests verify the capability to control the rate-dependence of materials with rate-dependent yield.

The tests consist of single-element simulations for a material with isotropic hardening and power law rate-dependent yield. Some of the tests use the native implementation of the power law rate-dependent yield, and others use an implementation using user subroutine *VUHARD*. *Table 1*, *Table 2*, *Table 3*, and *Table 4* list the material parameters. The test simulations ignore the rate-dependent yield. The simulation used as a reference is identical to the test simulations except that it uses the corresponding material definition with rate-independent yield.

Material:

Table 1: Density.

Density ρ	8.0×10^{-9}

Table 2: Elastic properties.

Young's modulus E	200.0×10^{3}
Poisson's ratio ν	0.3

Table 3: Hardening curve.

Yield stress	Plastic strain
550.00	0.0
650.00	0.01

Yield stress	Plastic strain
700.00	0.02
740.00	0.03
770.00	0.04
790.00	0.05
860.00	0.075
900.00	0.1

Table 4: Power law rate-dependent yield.

D	0.5
n	3.0

(The units are not important.)

Results and discussion

The simulations ignoring rate-dependent yield agree with the simulation with rate-independent yield and differ from the simulation that considers rate-dependent yield. *Figure 3* shows the comparison for the native implementation, and *Figure 4* shows the implementation using user subroutine *VUHARD*.

Input files

Abaqus/Standard input files

mises_plrate_ratedepoff.inp	Test ignoring rate-dependent yield.

Abaqus/Explicit input files

x_mises_plrate_ratedepoff.inp	Test ignoring rate-dependent yield.
x_mises_rateindependent.inp	Test with rate-independent yield.
x_mises_plrate_ratedepon.inp	Test with rate-dependent yield.
x_mises_pl_vuhard_ratedepoff.inp	Test with user subroutine *VUHARD* ignoring rate-dependent yield.
x_mises_pl_vuhard_ratedepon.inp	Test with user subroutine *VUHARD* considering rate-dependent yield.
iso_harden_pl_vuhard.f	User subroutine *VUHARD* used in `x_mises_pl_vuhard_ratedepoff.inp` and `x_mises_pl_vuhard_ratedepon.inp`.

Figures

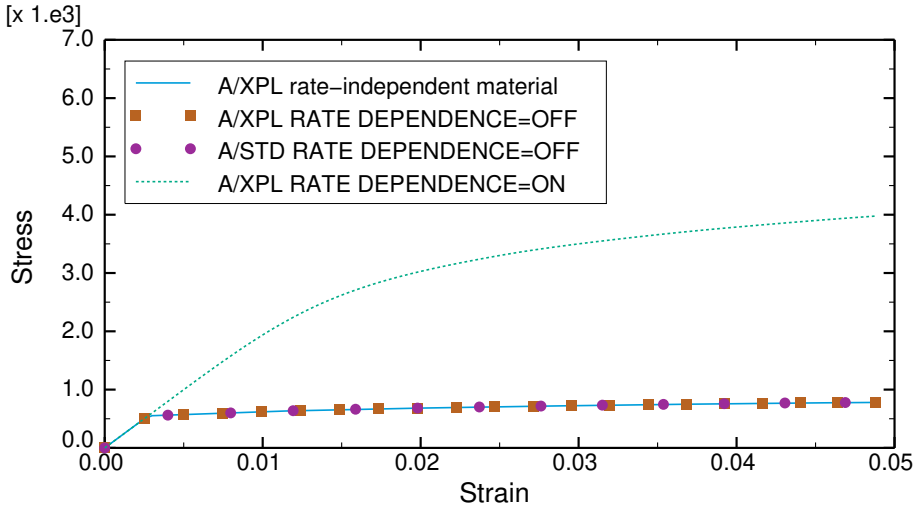


Figure 1: Rate-dependent and rate-independent response of a material with power law rate-dependent yield.

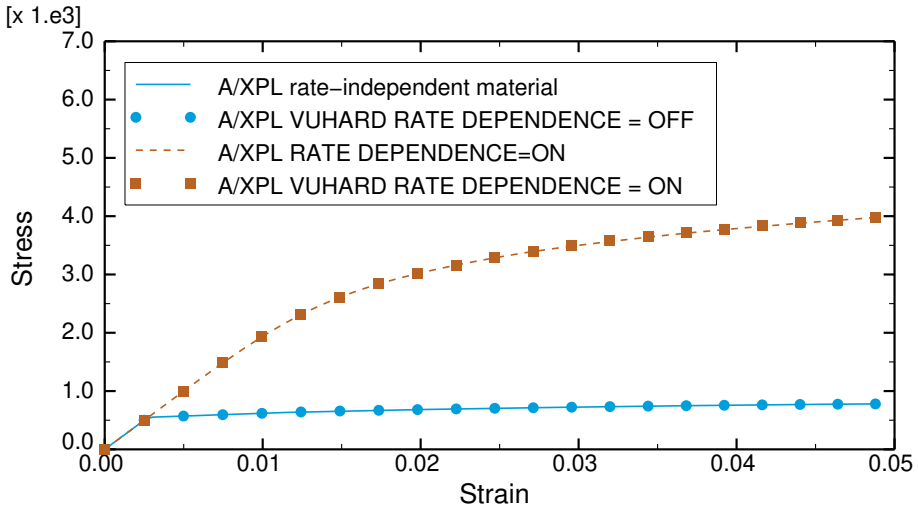


Figure 2: Rate-dependent and rate-independent response of a material with power law rate-dependent yield implemented using user subroutine `VUHARD`.

Viscoelasticity

Elements tested

C3D8 C3D8R

Features tested

Controlling the material rate-dependence of viscoelastic materials.

Problem description

The tests verify the capability to control the rate-dependence of viscoelastic materials.

The test simulations consist of single-element simulations for a neo-Hookean hyperelastic material with a one-term Prony linear viscoelasticity. *Table 5*, *Table 6*, and *Table 7* list the material parameters. These test simulations ignore the viscoelastic response and use the long-term material response. The simulation used as a reference is identical to the test simulations except that it uses the corresponding long-term material definition.

An analogous test verifies the capability for a PRF material with a nonlinear viscoelastic network with the modified Darveaux creep law. *Table 5*, *Table 6*, and *Table 8* list the material parameters. The simulation used as a reference is identical to the test simulation and considers the viscoelasticity in the material definition; however, the loading strain-rate is 500 times slower than in the test simulation to obtain the long-term response.

An analogous test verifies the capability for a PRF material with a nonlinear viscoelastic network with the Anand creep law. *Table 5*, *Table 6*, and *Table 9* list the material parameters. The simulation used as a reference is identical to the test simulation and considers the viscoelasticity in the material; however, the loading strain-rate is 500 times slower than in the test simulation to obtain the long-term response.

Material:

Table 5: Density.

Density ρ	1.0×10^{-5}

Table 6: Long-term neo-Hookean hyperelasticity.

C_{10}	8.0
D_1	0.1

Table 7: Prony linear viscoelasticity (N=1).

$\overline{g}_1^p$	0.5
$\overline{k}_1^p$	0.5
τ_1	1.0

Table 8: Modified Darveaux nonlinear creep.

C_{ss}	0.01
α	1.0
n	1.0
ϵ_T	1.0
B	1.0

Table 9: Anand nonlinear creep.

S_1	5.0

A	1.0
ξ	1.0
m	1.0
A_0	0.1
$\hat{s}$	0.2
n	0.5
a	1.3
S_2	0.0
S_3	0.0
A_1	0.0
A_2	0.0
A_3	0.0
A_4	0.0

(The units are not important.)

Results and discussion

The simulations ignoring the viscoelastic response agree with the simulation with the long-term material response and differ from the simulation considering the viscoelastic response.

Figure 5 shows the comparison for the hyperelastic material with linear viscoelasticity.

Figure 6, *Figure 7*, and *Figure 8* show the comparisons for the stress, network equivalent creep strain, and network steady-state equivalent creep strain for the PRF material with the modified Darveaux creep law.

Figure 9, *Figure 10*, and *Figure 11* show the comparisons for the stress, network equivalent creep strain, and network steady-state equivalent creep strain for the PRF material with the Anand creep law. The deformation resistance when the viscoelastic response is ignored only agrees with the deformation resistance from the simulation with the slow loading as time becomes infinitely large.

As expected, *Figure 5*, *Figure 6*, and *Figure 9* show that the stress responses of the three materials are identical when viscoelasticity is ignored.

Input files

Abaqus/Standard input files

nh_visco_longterm.inp	Test for the hyperelastic material with linear viscoelasticity ignoring viscoelasticity and using the long-term material response.
prf_moddarveaux_longterm.inp	Test for the PRF material with the modified Darveaux creep law ignoring viscoelasticity and using the long-term material response.
prf_anand_longterm.inp	Test for the PRF material with the Anand creep law ignoring viscoelasticity and using the long-term material response.

Abaqus/Explicit input files

x_nh_visco_ratedepoff.inp	Test for the hyperelastic material with linear viscoelasticity ignoring the viscoelasticity and using the long-term material response.
x_nh_rateindependent.inp	Test with the long-term hyperelastic material response and without viscoelasticity in the material definition.
x_nh_visco_ratedepon.inp	Test for the hyperelastic material considering the linear viscoelasticity in the material definition.
x_prf_moddarveaux_ratedepoff.inp	Test for the PRF material with the modified Darveaux creep law ignoring the nonlinearity viscoelasticity in the material definition and using the long-term material response.
x_prf_moddarveaux_slow.inp	Test for the PRF material with the modified Darveaux creep law considering the nonlinear viscoelasticity in the material definition but with a slow loading to obtain the long-term material response.
x_prf_moddarveaux_ratedepon.inp	Test for the PRF material with the modified Darveaux creep law considering the nonlinear viscoelasticity in the material definition.
x_prf_anand_ratedepoff.inp	Test for the PRF material with the Anand creep law ignoring the nonlinearity viscoelasticity in the material definition and using the long-term material response.
x_prf_anand_slow.inp	Test for the PRF material with the Anand creep law considering the nonlinear viscoelasticity in the material definition but with a slow loading to obtain the long-term material response.
x_prf_anand_ratedepon.inp	Test for the PRF material with the Anand creep law considering the nonlinear viscoelasticity in the material definition.

Figures

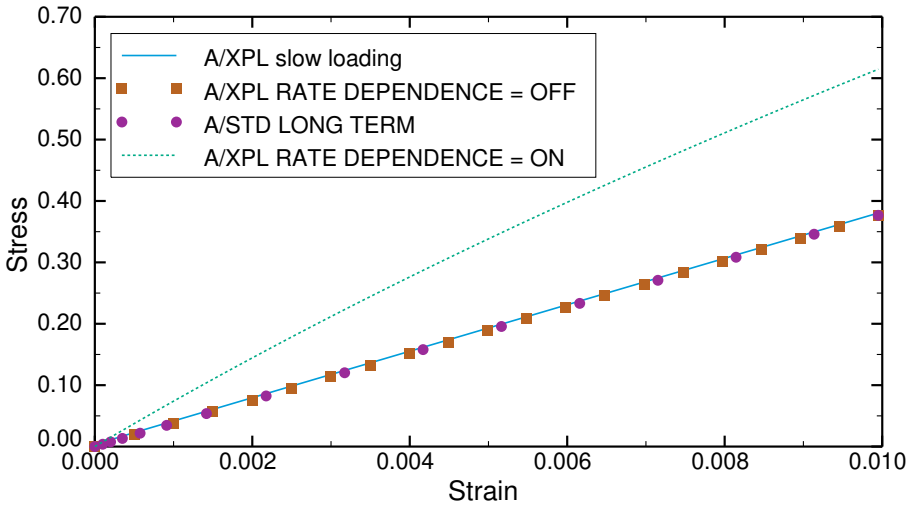


Figure 3: Rate-dependent and rate-independent response of a hyperelastic material with linear viscoelasticity.

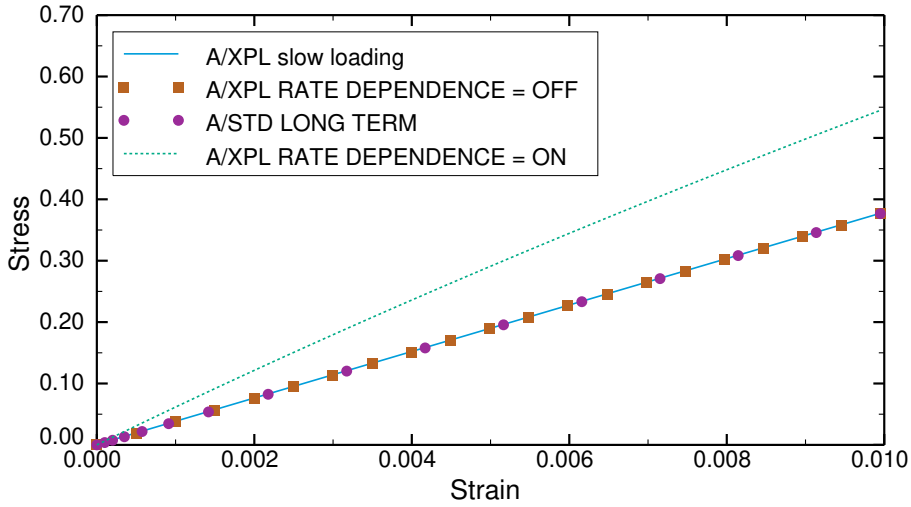


Figure 4: Rate-dependent and rate-independent stress response of a PRF material with the modified Darveaux creep law.

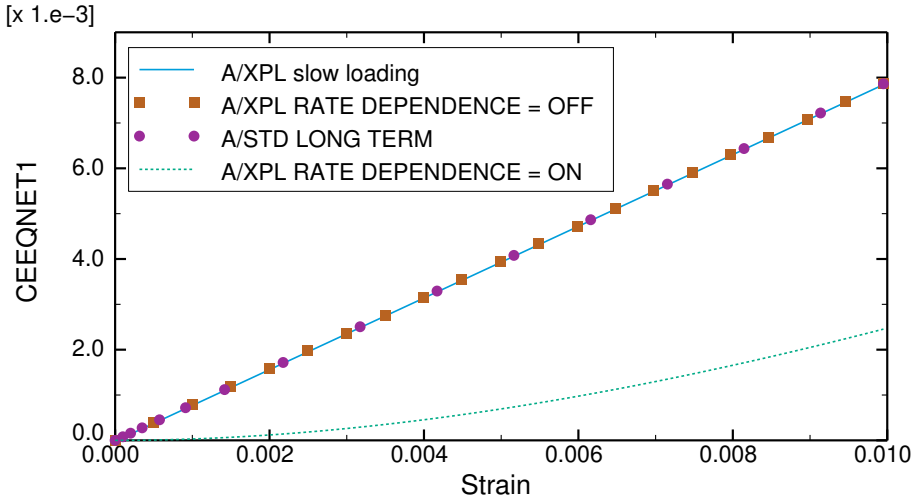


Figure 5: Rate-dependent and rate-independent network equivalent creep strain response of a PRF material with the modified Darveaux creep law.

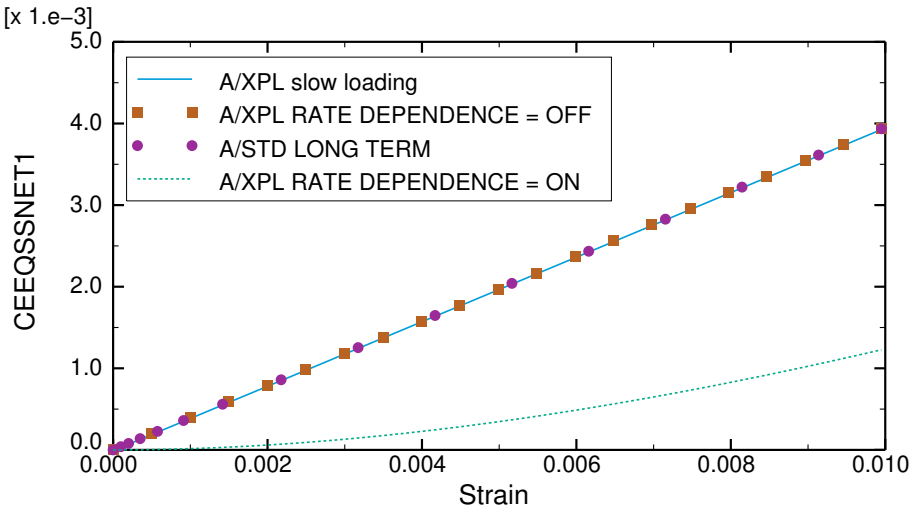


Figure 6: Rate-dependent and rate-independent network steady-state equivalent creep strain response of a PRF material with the modified Darveaux creep law.

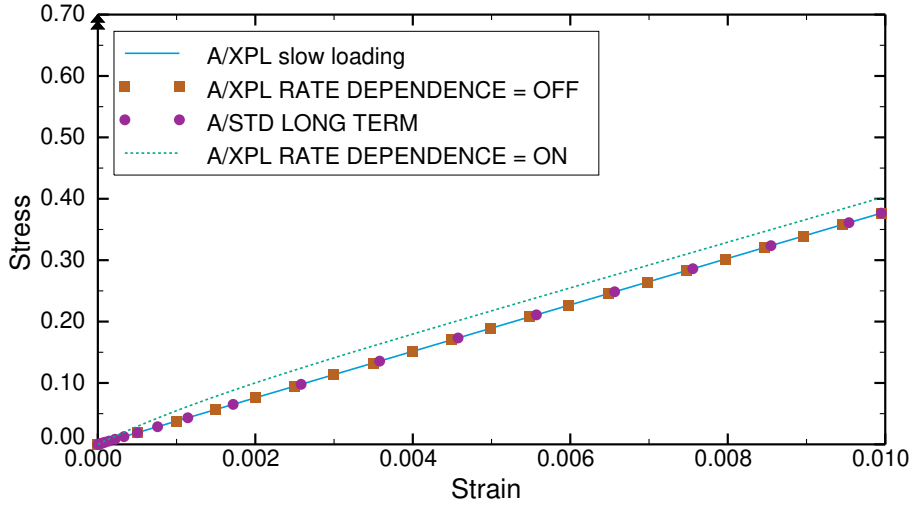


Figure 7: Rate-dependent and rate-independent stress response of a PRF material with the Anand creep law.

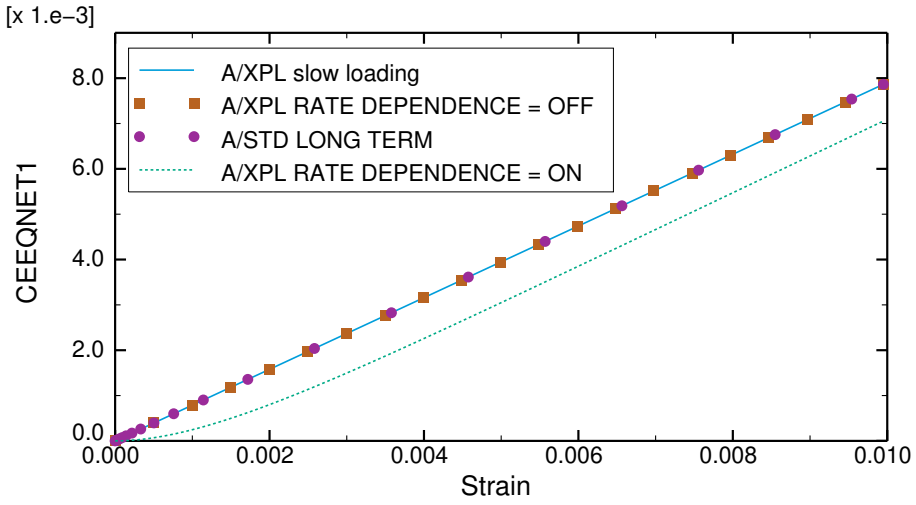


Figure 8: Rate-dependent and rate-independent network equivalent creep strain response of a PRF material with the Anand creep law.

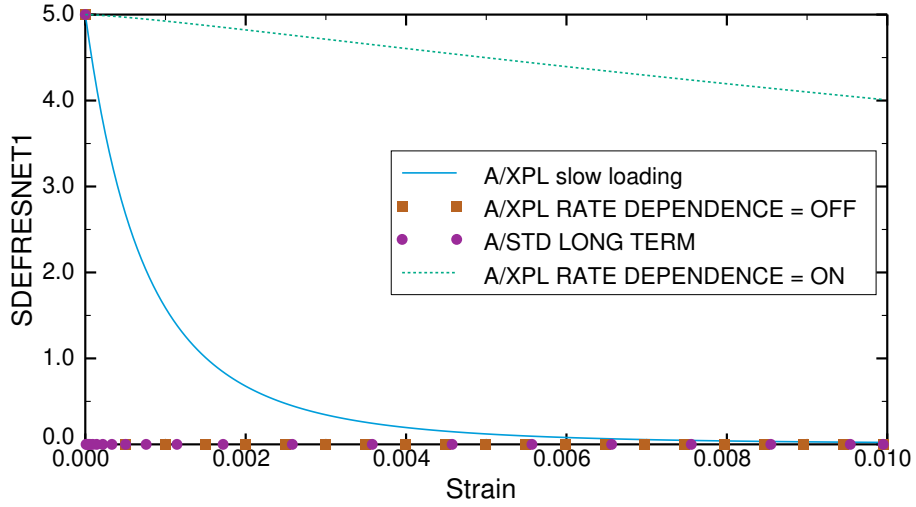


Figure 9: Rate-dependent and rate-independent network deformation resistance response of a PRF material with the Anand creep law.

Constraints Verification

This section defines the basic tests that are used to provide evidence that the implementation of the constraint options in Abaqus produces the expected results.

In this section:

- *Boundary conditions*
- *Multi-point constraints*
- *Surface-based tie constraint*
- *Rigid bodies*
- *Coupling constraints*
- *Kinematic coupling constraint*
- *Shell-to-solid coupling constraints*
- *Coupling of acoustic and structural elements*
- *Mesh-independent spot welds*
- *Embedded element technique*
- *Miscellaneous*

Boundary conditions

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Features tested

Various types of prescribed boundary conditions are tested.

Complex boundary conditions

Elements tested

AC2D4 CPS4

Problem description

The application of real and imaginary boundary conditions is tested in the direct-solution steady-state dynamic procedure. The test is performed in a structural analysis and an acoustic analysis. Each test is performed in three steps. The first step applies nonzero real boundary conditions to particular degrees of freedom of the structure, and the steady-state harmonic response is obtained. The second step is identical to the first step except that the nonzero boundary conditions are applied to the imaginary components of the specified degrees of freedom. The expected result is that the response of the degrees of freedom for the two steps should be identical but 90° out of phase from one another. The third step is identical to the first two steps except that nonzero boundary conditions are applied to both the real and imaginary components of the specified degrees of freedom. The expected result for this step is that the response of the degrees of freedom are 45° out of phase from the response in the previous two steps.

Results and discussion

Table 1: Complex boundary conditions, structural analysis (xbccplxs.inp).

	Frequency	$U1_1$	$U2_1$	$PU1_1$	$PU2_1$
Step 1:	50.0	0.0508	1.110	180.0	0.0
	100.0	0.0034	1.110	180.0	0.0
Step 2:	50.0	0.0508	1.110	−90.0	90.0
	100.0	0.0034	1.110	−90.0	90.0
Step 3:	50.0	0.0719	1.570	−135.0	45.0
	100.0	0.0048	1.570	−135.0	45.0

Table 2: Complex boundary conditions, acoustic analysis (xbccplxa.inp).

	Frequency	POR_1	$PPOR_1$	POR_{21}	$PPOR_{21}$
Step 1:	50.0	1.110	0.0	0.0546	180.0
	100.0	1.110	0.0	0.0126	180.0
Step 2:	50.0	1.110	90.0	0.0546	−90.0
	100.0	1.110	90.0	0.0126	−90.0
Step 3:	50.0	1.570	45.0	0.0772	−135.0
	100.0	1.570	45.0	0.0179	−135.0

Input files

xbccplxs.inp	Complex boundary conditions, structural analysis.
xbccplxa.inp	Complex boundary conditions, acoustic analysis.

Displacement, velocity, and acceleration boundary conditions

Elements tested

B21

Problem description

The input file *xbctypex.inp* tests the continuity of boundary conditions in a multistep dynamic analysis. The specifications of the boundary conditions are modified between steps. The displacement, velocity, and acceleration history are varied extensively to ensure proper transitions. The fixed boundary condition is tested to ensure that proper definitions are used to set the displacements at the respective nodal positions. In addition, the specifications for the boundary conditions are varied from user-specified amplitudes to user subroutine *DISP* to fixed boundary condition types (i.e., encastre, etc.) and even to the removal of the boundary condition specification altogether.

Results and discussion

Several combinations of boundary conditions are tested on several different nodes in this test. The boundary condition specifications on degree of freedom 2 of node 6 are discussed as a typical example.

The first step defines a constant acceleration of zero as the boundary condition for degree of freedom 2 of node 6. In doing so, the default amplitude variations for dynamic analysis are employed. As described in *Defining an Analysis*, the default choice of amplitude for a dynamic analysis is a step function except for the case of prescribed displacement and rotation boundary conditions, for which the default is a linear ramp function. Node 6 also has an initial velocity of 100. The resulting velocity and displacement should be integrated based on the prescribed acceleration variation including the initial velocity.

In the second step the specification is changed to velocity history and the step amplitude is applied linearly. Thus, the velocity should be linear from the previous value at the end of the first step to the final value (0.0) set in the definition for this step. The resulting displacement and acceleration histories should reflect this prescribed variation.

In the third step the step amplitude is applied linearly, and the boundary specification is changed to reference user subroutine *DISP*. In the user subroutine the acceleration is the value of the magnitude factor, which is ramped over the step. The velocity and displacements are the appropriate integrals of this variation. Since the amplitude is applied linearly, the magnitude factor is ramped during this step from the previous displacement

value of 100 to the final value of 10 given in the boundary condition definition for this step. This linear definition modifies the function specified in the user subroutine such that the acceleration is linear, the velocity is quadratic, and the displacement is cubic. The curves for this typical boundary condition specification are given in *Figure 1*. Many other variations of boundary condition specifications are verified in the test.

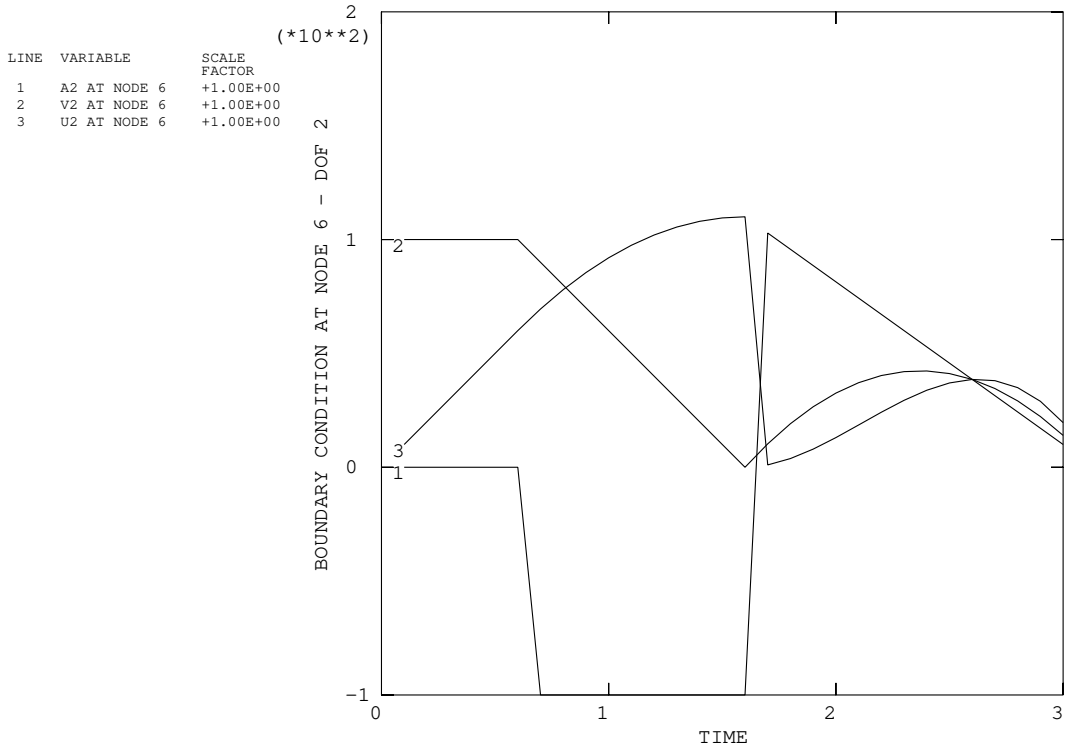


Figure 1: Boundary condition history for node 6, degree of freedom 2.

Input files

xbctypex.inp	TYPE boundary conditions.
xbctypex.f	User subroutine *DISP* used in xbctypex.inp.

Velocity-type boundary conditions, static analysis

Elements tested

B21

Problem description

Input file *xbcvelstat.inp* tests the continuity of boundary conditions (primarily velocities) as they are modified between steps in a multistep static analysis. The velocities are always known in dynamic analysis, but they are not calculated and stored during a static analysis. Therefore, the use of velocity specifications in static analysis presents some unique problems.

Input files *xbcvelres1.inp* and *xbcvelres2.inp* test the restart capability for velocity-type boundary conditions when used in a static analysis. *xbcvelres1.inp* does not terminate the current step in the analysis from which the restart is made, while *xbcvelres2.inp* does. The input files are designed such that the results from the restart analyses are the same as those from the original analysis.

Results and discussion

The original analysis consists of four static steps using beam elements. The nine available degrees of freedom are exercised through modifications of the boundary condition specifications between steps. Boundary conditions that are specified as velocities with an amplitude reference in one step can be modified to a displacement specification with a ramped amplitude specification in the next or can be fixed or removed altogether. The continuity of the boundary conditions, when examined, is seen to be correct. The restart analyses produce results that are identical to those in the original analysis.

Input files

xbcvelstat.inp	TYPE=VELOCITY boundary conditions, static analysis.
xbcvelres1.inp	**RESTART* without END STEP test for xbcvelstat.inp.
xbcvelres2.inp	**RESTART* with END STEP test for xbcvelstat.inp.

Multi-point constraints

Products: Abaqus/Standard Abaqus/Explicit

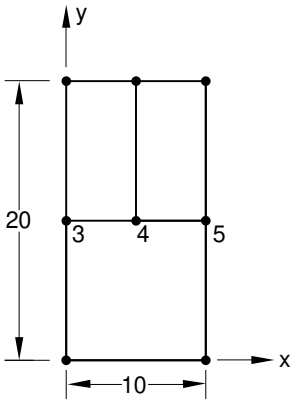
Overview

This problem contains basic test cases for one or more Abaqus elements and features.

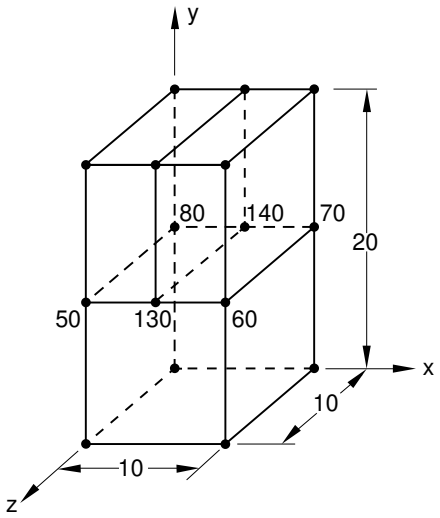
Features tested

Various types of multi-point constraints are tested. Simple geometries are given displacements or loads that result in easily checked responses. These responses confirm the proper functioning of the MPCs being tested. Unless noted otherwise, the static procedure is tested. All explicit dynamic tests have been performed so that a quasi-static solution is obtained.

LINEARMPC



LINEAR,4,3,5



LINEAR,130,50,60
LINEAR,140,80,70

The LINEARMPC is tested in Abaqus/Standard and Abaqus/Explicit. A cantilevered bar is subjected to a uniform tensile loading on the free end.

Abaqus/Standard analysis

Elements tested

C3D8 CPS4

Problem description

Model: Two models (one consisting of CPS4 elements and the other consisting of C3D8 elements) were created within one input file.

Material: Linear elastic, Young's modulus = 3.0×10^6, Poisson's ratio = 0.3.

Boundary conditions: u_x=0 at x=0, u_y=0 at y=0, and u_z=0 at z=0 for three-dimensional models.

Loading:

- Step 1: A uniform pressure of 10000 in the y-direction is applied to the top surface.
- Step 2: The load that was applied in the first step is applied again, this time using NLGEOM for large-displacement analysis.

Results and discussion

The results obtained agree with the analytical solution.

Input files

xmpcline.inp	LINEARMPC.
xmpclinet.inp	LINEARMPC with transforms.

Abaqus/Explicit analysis

Elements tested

C3D8R CPS4R

Problem description

Model: Two models (one consisting of CPS4R elements and the other consisting of C3D8R elements) were created within one input file.

Material: Linear elastic, Young's modulus = 3.0×10^6, Poisson's ratio = 0.3, density = 0.03.

Boundary conditions: $u_x = 0$ at $x = 0$, $u_y = 0$ at $y = 0$, and $u_z = 0$ at $z = 0$ for three-dimensional models.

Loading: A uniform pressure of 10000 in the y-direction is applied to the top surface.

Results and discussion

The expected solution variables are obtained, and compatibility in the displacement solutions is observed.

Input files

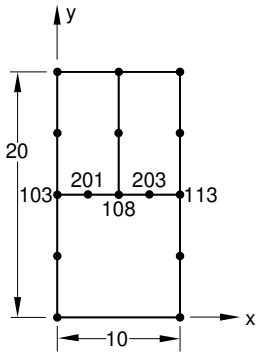
mpc_linear.inp	Input data for this MPC test.

QUADRATIC, BILINEAR, and C BIQUADMPCs

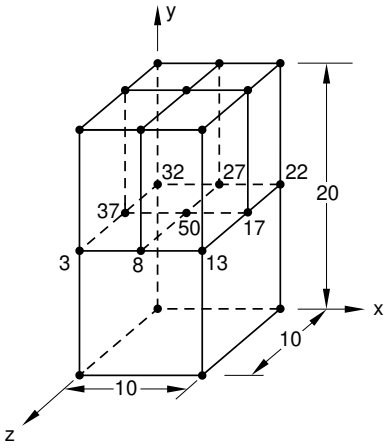
Elements tested

C3D8 C3D20 CPS8

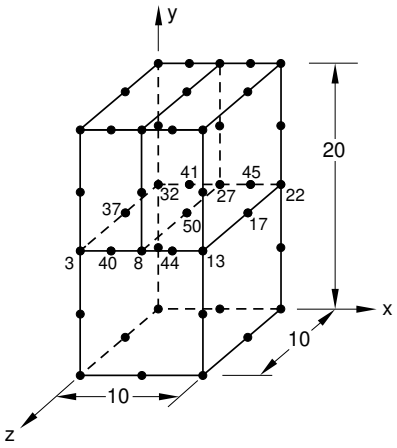
Problem description



QUADRATIC,201,103,108,113
QUADRATIC,203,103,108,113



LINEAR,8,3,13
LINEAR,17,13,22
LINEAR,27,22,32
LINEAR,37,32,3
BILINEAR,50,3,13,22,32



QUADRATIC,40,3,8,13
QUADRATIC,44,3,8,13
QUADRATIC,45,22,27,32
QUADRATIC,41,22,27,32
C BIQUAD,50,3,13,22,32,8,17,27,37

The QUADRATIC, BILINEAR, and C BIQUADMPCs are tested in Abaqus/Standard. A cantilevered bar is subjected to a uniform tensile loading on the free end.

The following model data apply to all three tests:

Material: Linear elastic, Young's modulus = 3.0×10^6, Poisson's ratio = 0.3.

Boundary conditions: u_x=0 at x=0, u_y=0 at y=0, and u_z=0 at z=0 for three-dimensional models.

Loading:

- Step 1: A uniform pressure of 10000 in the *y*-direction is applied to the top surface.
- Step 2: The load that was applied in the first step is applied again, this time using NLGEOM for large-displacement analysis.

Results and discussion

The results obtained agree with the analytical solution.

Input files

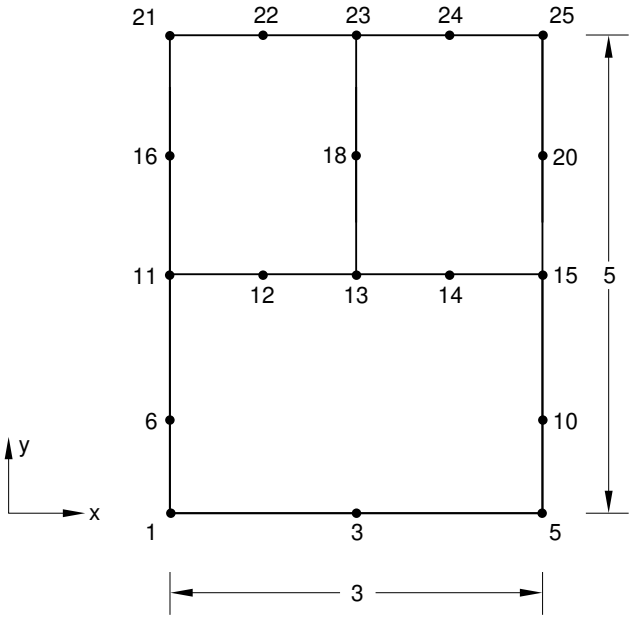
xmpcquad.inp	QUADRATICMPC.
xmpcquadt.inp	QUADRATICMPC with transforms.
xmpcbili.inp	BILINEAR and LINEARMPCs; MPC data read from input file xmpcinfo.inp.
xmpcbilit.inp	BILINEAR and LINEARMPCs with transforms; MPC data read from input file xmpcinfo.inp.
xmpccbiq.inp	C BIQUAD and QUADRATICMPCs.
xmpccbiqt.inp	C BIQUAD and QUADRATICMPCs with transforms.

P LINEARMPC

Elements tested

CPE8P

Problem description



P LINEAR,13,11,15
QUADRATIC,14,11,13,15
QUADRATIC,12,11,13,15

The P LINEARMPC is tested in Abaqus/Standard.

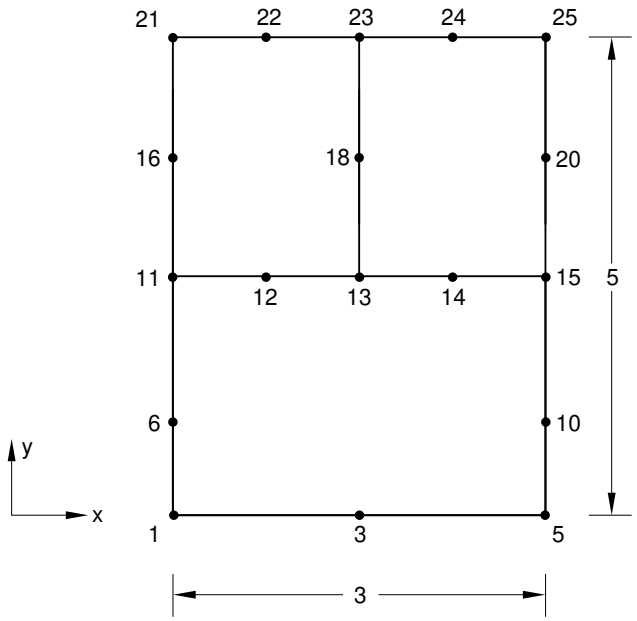
Boundary conditions: All displacement degrees of freedom are restrained throughout the analysis. In Step 1 the pore pressure is set to zero at nodes 1 and 5. In Step 2 the pore pressure is set to zero at nodes 5, 15, and 25.

Loading:

- Step 1: A pore fluid velocity is specified along the top of the model.
- Step 2: A pore fluid velocity is specified along the left edge of the model.

Results and discussion

The results obtained agree with the analytical solution.



P LINEAR,13,11,15
QUADRATIC,14,11,13,15
QUADRATIC,12,11,13,15

Input files

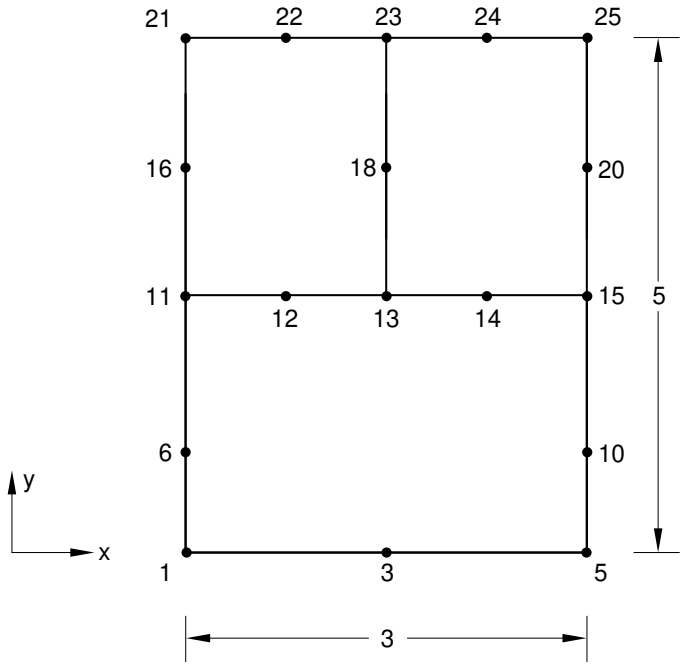
xmpcplin.inp P LINEAR and QUADRATICMPCs.

T LINEARMPC

Elements tested

CPE8T CPEG8T

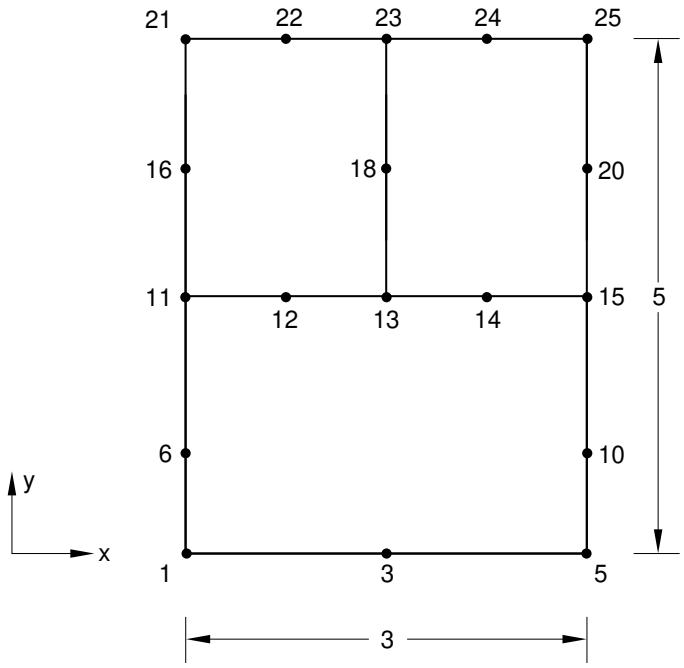
Problem description




```
T LINEAR,13,11,15
QUADRATIC,12,11,13,15
QUADRATIC,14,11,13,15
```

The T LINEARMPC is tested in Abaqus/Standard.

Boundary conditions: All displacement degrees of freedom are restrained throughout the analysis. In Step 1 the temperature is set to zero at nodes 5, 15, and 25. In Step 2 the temperature is set to zero at nodes 1 and 5.



T LINEAR,13,11,15
QUADRATIC,12,11,13,15
QUADRATIC,14,11,13,15

Loading:

- Step 1: A film coefficient and sink temperature are specified along the left edge of the model.
- Step 2: An emissivity and sink temperature are specified along the top edge of the model.

Results and discussion

The results obtained agree with the analytical solution.

Input files

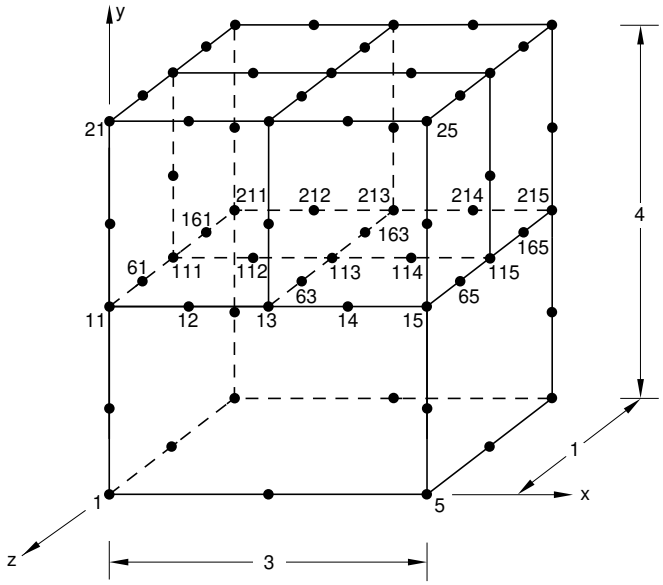
xmpctlin.inp	T LINEAR and QUADRATICMPCs.
xmpctlin_cpeg8t.inp	T LINEAR and QUADRATICMPCs.

P BILINEARMPC

Elements tested

C3D20P

Problem description



P BILINEAR,113,11,15,215,211
P LINEAR,13,11,15
P LINEAR,115,15,215
P LINEAR,213,211,215
P LINEAR,111,11,211
C BIQUAD,63,11,15,115,111,13,65,113,61
C BIQUAD,114,13,15,215,213,14,115,214,113
C BIQUAD,163,111,115,215,211,113,165,213,161
C BIQUAD,112,11,13,213,211,12,113,212,111
QUADRATIC,65,15,115,215
QUADRATIC,165,15,115,215
QUADRATIC,61,11,111,211
QUADRATIC,161,11,111,211
QUADRATIC,12,11,13,15
QUADRATIC,14,11,13,15
QUADRATIC,212,211,213,215
QUADRATIC,214,211,213,215

The P BILINEARMPC is tested in Abaqus/Standard.

Boundary conditions: All displacement degrees of freedom are restrained throughout the analysis. In Step 1 the pore pressure is set to zero on the front face of the model. In Step 2 the pore pressure is set to zero on the right face of the model.

Loading:

- Step 1: A pore fluid velocity is specified out of the back face of the model.
- Step 2: A pore fluid velocity is specified out of the left face of the model.

Results and discussion

The results obtained agree with the analytical solution.

Input files

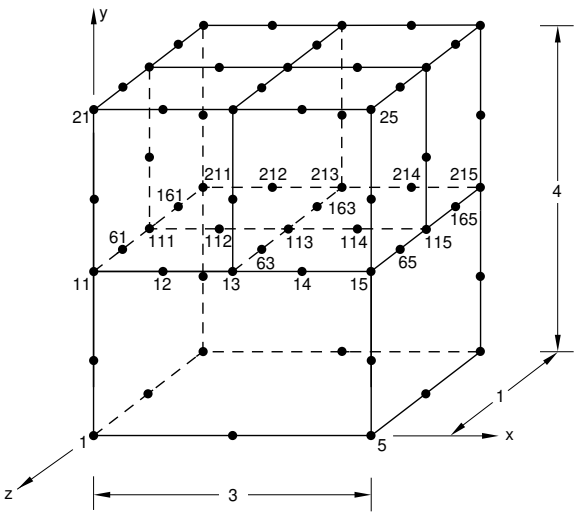
xmpcpbil.inp	P BILINEAR, P LINEAR, C BIQUAD and QUADRATICMPCs.

T BILINEARMPC

Elements tested

C3D20T

Problem description




```
T BILINEAR,113,11,15,215,211
T LINEAR,13,11,15
T LINEAR,115,15,215
T LINEAR,213,211,215
T LINEAR,111,11,211
C BIQUAD,63,11,15,115,111,13,65,113,61
C BIQUAD,114,13,15,215,213,14,115,214,113
C BIQUAD,163,111,115,215,211,113,165,213,161
C BIQUAD,112,11,13,213,211,12,113,212,111
QUADRATIC,65,15,115,215
QUADRATIC,165,15,115,215
QUADRATIC,61,11,111,211
QUADRATIC,161,11,111,211
QUADRATIC,12,11,13,15
QUADRATIC,14,11,13,15
QUADRATIC,212,211,213,215
QUADRATIC,214,211,213,215
```

The T LINEARMPC is tested in Abaqus/Standard.

Boundary conditions: All displacement degrees of freedom are restrained throughout the analysis. In Step 1 the temperature is set to zero on the left face of the model. In Step 2 the temperature is set to zero on the front face of the model.

Loading:

- Step 1: An emissivity and sink temperature are given on the left face of the model.
- Step 2: A surface flux is specified on the back face of the model.

Results and discussion

The results obtained agree with the analytical solution.

Input files

xmpctbil.inp	T BILINEAR, T LINEAR, C BIQUAD and QUADRATICMPCs.

BEAMMPC

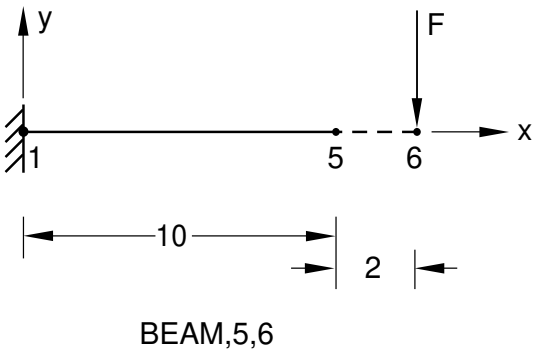
The BEAMMPC is tested in Abaqus/Standard and Abaqus/Explicit. A cantilevered beam is subjected to a transverse tip load.

Abaqus/Standard analysis

Elements tested

B22 B32

Problem description



Two-dimensional and three-dimensional beams are considered, with and without the RIKS procedure (introduces a slight imperfection corresponding to the first buckling mode).

Material: Linear elastic, Young's modulus = 3.0×10^6, Poisson's ratio = 0, density = 1700.

Boundary conditions: Node 1 is clamped.

Loading 1:

- Step 1: F_y=−1000 at node 3.
- Step 2: The first four buckling modes are extracted for a live load of F_x=−1000.
- Step 3: The load that was applied in the first step is applied again, this time using NLGEOM for large-displacement analysis.

Loading 2:

- Step 1: The first four buckling modes are extracted for a live load of F_x=−1.
- Step 2: A RIKS procedure is adopted until a maximum load of F_x=−300 at node 6.

Results and discussion

The results agree with the theoretically expected results. The results of the buckling analyses and the geometrically nonlinear analyses show that the initial stress terms are accounted for correctly.

Input files

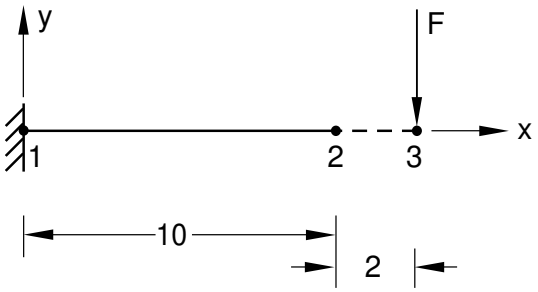
xmpcbeam.inp	Two-dimensional beam.
xmpcbeamt.inp	Two-dimensional beam with transforms.
xmpcbem3.inp	Three-dimensional beam.
ver_xmpcbem3_thermexp.inp	Three-dimensional beam with temperature increase.
xmpcbem3t.inp	Three-dimensional beam with transforms.
xmpcbemr.inp	Two-dimensional beam with RIKS.
xmpcbemrt.inp	Two-dimensional beam with RIKS and transforms.
xmpcbm3r.inp	Three-dimensional beam with RIKS.
xmpcbm3rt.inp	Three-dimensional beam with RIKS and transforms.

Abaqus/Explicit analysis

Elements tested

B31 MASS PIPE31

Problem description



BEAM, 2, 3

The following equivalent cases are considered:

1. A BEAM-type MPC is defined between nodes 2 and 3.
2. Nodes 2 and 3 are included in a rigid body tie-type node set.
3. Nodes 2 and 3 are connected by a beam element of type B31. This element is then included in a rigid body.

Material: Linear elastic, Young's modulus = 3.0×10^6, Poisson's ratio = 0, density = 0.03.

Boundary conditions: Node 1 is clamped.

Loading: F_y=−1000 at node 3.

Beam section data: B31, 1 × 1 rectangle. PIPE31, pipe of radius 1 and thickness 0.1.

Results and discussion

To verify that the MPC is working correctly, the rotation at node 3 should be the same as the rotation at node 2; the vertical displacement at node 3 should be given by $(u_y)_3 = (u_y)_2 + 2\sin\phi_z$. This solution is obtained. The results for Cases 2 and 3 match the results for Case 1.

Input files

mpc_beam.inp	Input data for Case 1 for beam elements.
mpc_beamrig1.inp	Input data for Case 2 for beam elements.
mpc_beamrig2.inp	Input data for Case 3 for beam elements.
mpc_beam_pipe.inp	Input data for Case 1 for pipe elements.
mpc_beamrig1_pipe.inp	Input data for Case 2 for pipe elements.
mpc_beamrig2_pipe.inp	Input data for Case 3 for pipe elements.

ELBOWMPC

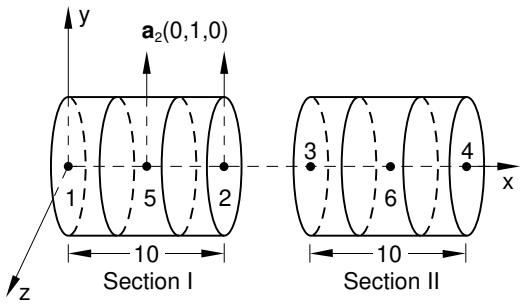
Elements tested

ELBOW31 ELBOW32

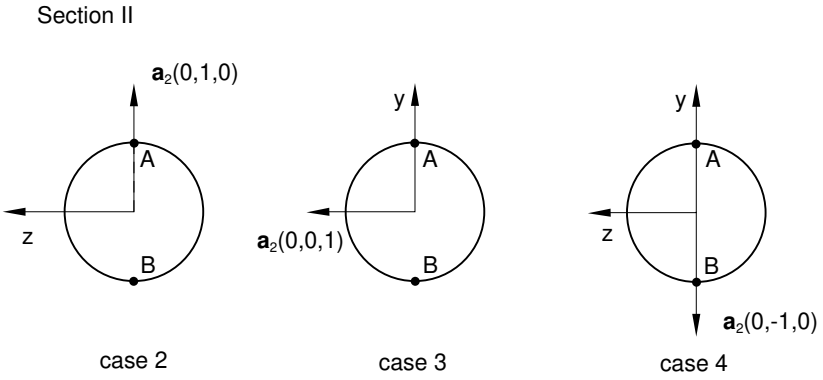
Problem description

The ELBOWMPC is tested in both static and dynamic analyses in Abaqus/Standard.

Four cases are tested with each element type in the static analyses (see *Figure 1*).



Node 5 and node 6 are only present in XMPCELB2



Note: case 1 has no MPC and no node 3

ELBOW,2,3

Figure 1: ELBOWMPC geometry.

In addition to the differences shown in the figure, there are the following differences:

- Case 1: Control model. No ELBOWMPC. Otherwise the same as Case 4.
- Case 2: 16 integration points around the pipe; 3 section points through the thickness; 5 Fourier ovalization modes.
- Case 3: 12 integration points around the pipe; 5 section points through the thickness; 4 Fourier ovalization modes.
- Case 4: 20 integration points around the pipe; 5 section points through the thickness; 6 Fourier ovalization modes.

The following data apply to the four cases in each file:

Boundary conditions:

Node 1 has degrees of freedom 1–6 fixed. All nodes have NODEFORM condition.

Loading:

- Step 1: $\boldsymbol{M_z}=1 \times 10^6$ at node 4.
- Step 2: $\boldsymbol{F_x}=2 \times 10^6$ at node 4.

- Step 3: The load that was applied in the first step is applied again, this time using NLGEOM for large-displacement analysis.
- Step 4: The load that was applied in the second step is applied again, this time using NLGEOM for large-displacement analysis.

General: Two straight pipes, each discretized with two elements, are considered in the dynamic analysis. In the first case the second cross-sectional directions of both elements are identical and the ELBOWMPC is not used. In the second case the second cross-sectional directions are different and the ELBOWMPC is used to ensure continuity of displacements. The analysis consists of two steps. In the static step the pipes are subjected to bending by applying a concentrated force. In the direct-integration implicit dynamic step the force is removed and the pipes vibrate freely.

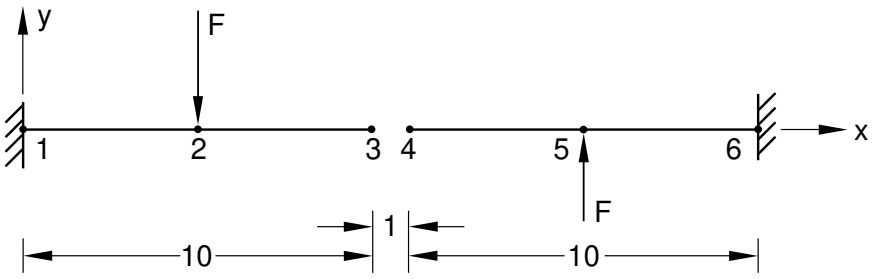
Results and discussion

For the static analyses Cases 2–4 give the same answer as Case 1; σ_{axial} at points A and B match. In the dynamic case the results for both pipes (with and without the ELBOWMPC) are identical.

Input files

xmpcelb1.inp	ELBOW31 elements; static analysis.
xmpcelb1t.inp	ELBOW31 elements; static analysis with transforms.
xmpcelb2.inp	ELBOW32 elements; static analysis.
xmpcelb2t.inp	ELBOW32 elements; static analysis with transforms.
xmpcelb3.inp	ELBOW31 elements; dynamic analysis.

LINKMPC



LINK,3,4

The LINKMPC is tested in Abaqus/Standard and Abaqus/Explicit. Two cantilevered beams are subjected to transverse loading.

Abaqus/Standard analyses

Elements tested

B23 B33

Problem description

Material: Linear elastic, Young's modulus = 3.0×10^6, Poisson's ratio = 0, density = 7800.0.

Boundary conditions: Nodes 1 and 6 are clamped.

Loading:

- Step 1: The first four natural frequencies are extracted.
- Step 2: F_y=–250 at node 2, F_y=250 at node 5.
- Step 3: The loads that were applied in the previous step are applied again, this time using NLGEOM for large-displacement analysis.

Results and discussion

The LINKMPC provides a pinned, rigid link between two nodes. For this example this means that the translational degrees of freedom should have equal magnitudes but opposite sense and the rotational degree of freedom should be the same for the nodes that are joined by the MPC. This solution is obtained.

Input files

xmpclink.inp	Two-dimensional beam.
xmpclinkt.inp	Two-dimensional beam with transforms.
xmpclnk3.inp	Three-dimensional beam.
ver_xmpclnk3_thermexp.inp	Three-dimensional beam with temperature increase.
xmpclnk3t.inp	Three-dimensional beam with transforms.

Abaqus/Explicit analyses

Elements tested

B31 PIPE31 ROTARYI T3D2

Problem description

The following equivalent cases are considered:

1. A LINK-type MPC is defined between nodes 3 and 4.
2. Nodes 3 and 4 are included in a rigid body pin-type node set.
3. Nodes 3 and 4 are connected by a truss element of type T3D2. This element is then included in a rigid body.

Material: Linear elastic, Young's modulus = 3.0×10^6, Poisson's ratio = 0, density = 0.03.

Boundary conditions: Nodes 1 and 6 are clamped.

Loading: F_y=–250 at node 2, F_y=250 at node 5.

Beam section data: B31, 1 × 1 rectangle. PIPE31, pipe of radius 1 and thickness 0.1.

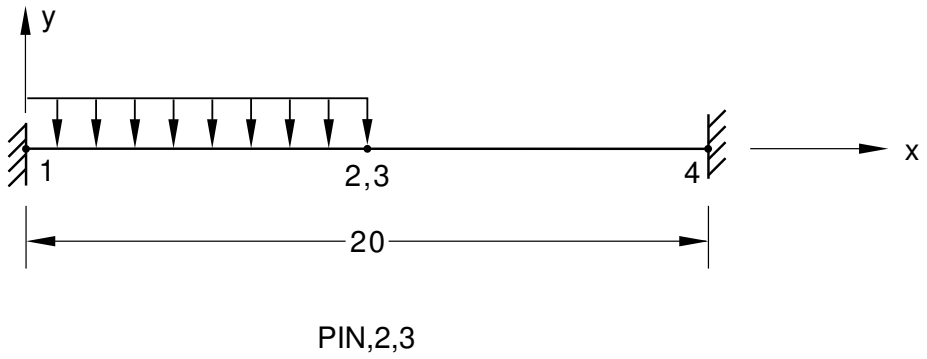
Results and discussion

The LINKMPC provides a pinned, rigid link between two nodes. For this example this means that the translational degrees of freedom should have equal magnitudes but opposite sense and the rotational degree of freedom should be the same for the nodes that are joined by the MPC. This solution is obtained. The results for Cases 2 and 3 match the results for Case 1.

Input files

mpc_link.inp	Input data for Case 1 for beam elements.
mpc_linkrig1.inp	Input data for Case 2 for beam elements.
mpc_linkrig2.inp	Input data for Case 3 for beam elements.
mpc_link_pipe.inp	Input data for Case 1 for pipe elements.
mpc_linkrig1_pipe.inp	Input data for Case 2 for pipe elements.
mpc_linkrig2_pipe.inp	Input data for Case 3 for pipe elements.

PINMPC



PIN,2,3

The PINMPC is tested in Abaqus/Standard and Abaqus/Explicit. A beam structure that is cantilevered at both ends has a pressure loading applied to one-half of the model.

Abaqus/Standard analysis

Elements tested

B23

Problem description

Material: Linear elastic, Young's modulus = 3.0×10^6, Poisson's ratio = 0.

Boundary conditions: Nodes 1 and 4 are clamped.

Loading:

- Step 1: The left half of the beam is loaded by a force per unit length, PY=−1000.
- Step 2: The load that was applied in the first step is applied again, this time using NLGEOM for large-displacement analysis.

Beam section data: B23, 1 × 1 rectangle.

Results and discussion

The PINMPC provides a pinned joint between two nodes by making the translational degrees of freedom equal. The displacements of nodes 2 and 3 are identical.

Input files

xmpcpinx.inp	PINMPC.
xmpcpinxt.inp	PINMPC with transforms.

Abaqus/Explicit analyses

Elements tested

B21 PIPE21 ROTARYI

Problem description

The following equivalent cases are considered:

1. A PIN-type MPC is used to connect nodes 2 and 3.
2. Nodes 2 and 3 are included in a rigid body pin-type node set.

Material: Linear elastic, Young's modulus = 3.0×10^6, Poisson's ratio = 0, density = 0.03.

Boundary conditions: Nodes 1 and 4 are clamped.

Loading: The left half of the beam is loaded by a force per unit length, PY=−1000.

Beam section data: B21, 1 × 1 rectangle.

PIPE21, pipe of radius 1 and thickness 0.1.

Results and discussion

The PINMPC provides a pinned joint between two nodes by making the translational degrees of freedom equal. The displacements of nodes 2 and 3 are identical. The results for Case 2 match the results for Case 1.

Input files

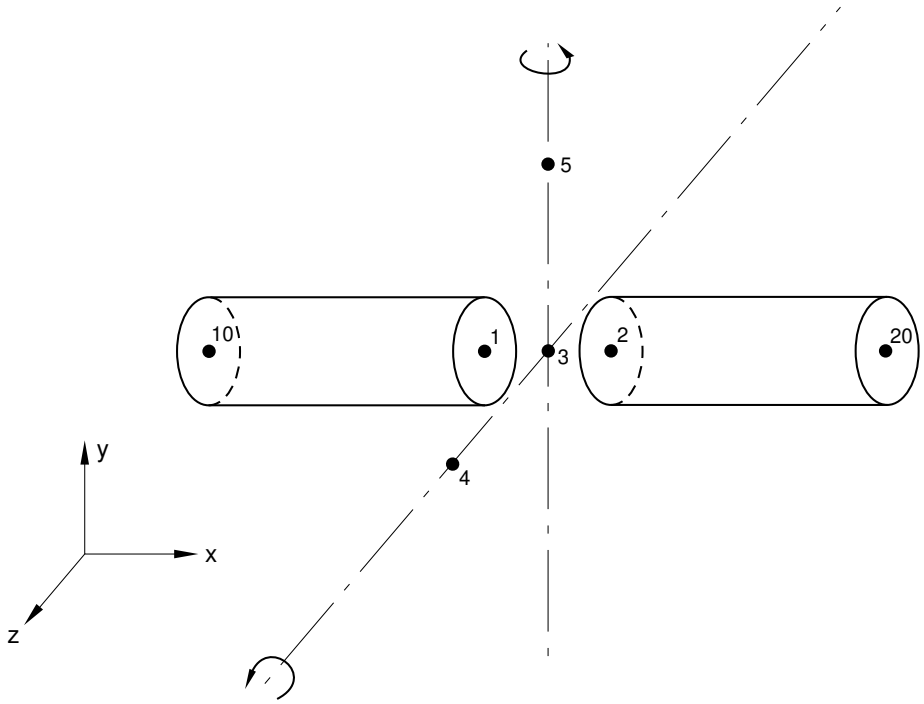
mpc_pin.inp	Input data for Case 1 for beam elements.
mpc_pinrig.inp	Input data for Case 2 for beam elements.
mpc_pin_pipe.inp	Input data for Case 1 for pipe elements.
mpc_pinrig_pipe.inp	Input data for Case 2 for pipe elements.

REVOLUTEMPC

Elements tested

B33H

Problem description



PIN,1,2
PIN,2,3
REVOLUTE,2,3,5
REVOLUTE,3,1,4

The REVOLUTEMPC is tested in Abaqus/Standard.

Boundary conditions: All degrees of freedom are restrained at node 10 throughout the analysis. Nodes 5 and 6 are initially constrained in degree of freedom 6.

Loading:

- Step 1: A concentrated follower force is applied at node 20 to pull the joint.
- Step 2: The joint is rotated by 45° about the 3–4 joint axis by prescribing degree of freedom 6 at node 4.
- Step 3: The joint is rotated by 45° about the current 3–5 axis by prescribing degree of freedom 6 at node 5.

Results and discussion

The axial follower force of Step 1 couples with the rotations in subsequent steps to cause a lateral deflection of node 1 despite a very high material modulus.

Input files

xmpcrevo.inp	REVOLUTE and PINMPCs.
xmpcrevot.inp	REVOLUTE and PINMPCs with transforms.

SLIDERMPC

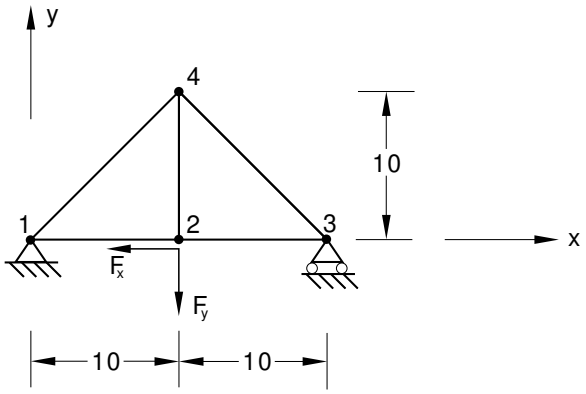
The SLIDERMPC is tested in Abaqus/Standard for a truss and a beam structure and in Abaqus/Explicit for a truss structure.

Abaqus/Standard truss analyses

Elements tested

T2D2

Problem description



SLIDER,2,1,3

A truss structure has a SLIDERMPC connecting node 2 to nodes 1 and 3.

Material: Linear elastic, Young's modulus = 3.0×10^6, Poisson's ratio = 0.

Boundary conditions: u_x=u_y=0 at node 1, u_y=0 at node 3.

Load case 1:

- Step 1: F_x=−500 at node 2, F_y=−1000 at node 2.
- Step 2: The loads that were applied in the first step are applied again, this time using NLGEOM for large-displacement analysis.

Load case 2: F_x=−500 at node 2, F_y=−1000 at node 2. A static Riks step is adopted.

Truss section data: T2D2, cross-sectional area = 1.

Results and discussion

The SLIDERMPC keeps a node on a straight line between two nodes but allows it to slide along the line and the line to change length. This solution is obtained. The geometrically nonlinear analyses show that the initial stress terms are accounted for correctly.

Input files

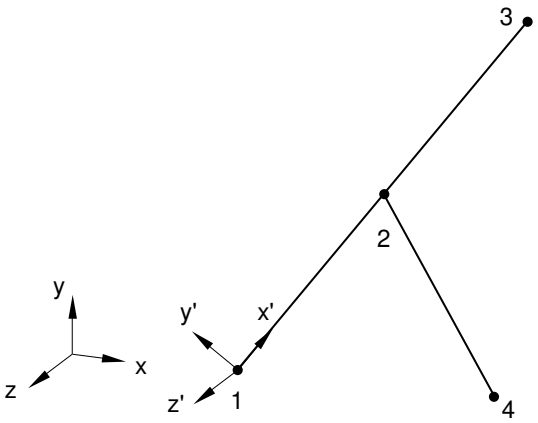
xmpcslid.inp	SLIDERMPC.
xmpcslidt.inp	SLIDERMPC with transforms.
xmpcsldr.inp	SLIDERMPC with RIKS.
xmpcsldrt.inp	SLIDERMPC with RIKS and transforms.

Abaqus/Standard beam analysis

Elements tested

B31

Problem description



SLIDER,2,1,3

Material: Linear elastic, Young's modulus = 3.0×10^6, Poisson's ratio = 0.

Boundary conditions: $u_x=u_y=u_z=\phi_z=0$ at node 4. All displacements and rotations are fixed at node 1. A transformation at node 1 places the local x-axis along the direction from node 1 to node 3.

Loading:

- Step 1: F_y=10 at node 3. Node 1 is rotated about the transformed z-axis. (ϕ_z=0.3.)
- Step 2: The load and displacement that were applied in the first step are applied again, this time using NLGEOM for large-displacement analysis.

Beam section data: B31, cross-sectional area = 1.

Results and discussion

The SLIDERMPC keeps a node on a straight line between two nodes but allows it to slide along the line and the line to change length. This solution is obtained. The geometrically nonlinear analyses show that the initial stress terms are accounted for correctly.

Input files

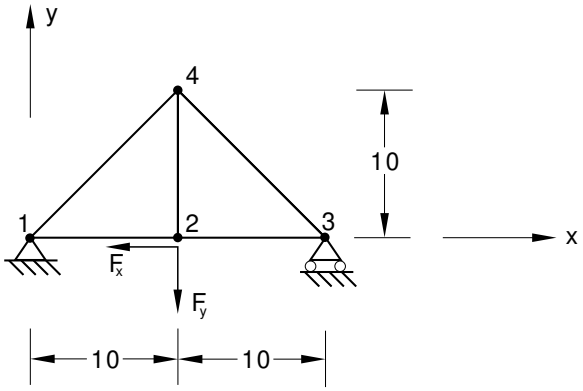
xmpcsld3.inp	SLIDERMPC.
xmpcsld3t.inp	SLIDERMPC with transforms.

Abaqus/Explicit analysis

Elements tested

T2D2

Problem description



SLIDER, 2, 1, 3

A truss structure has a SLIDERMPC connecting node 2 to nodes 1 and 3.

Material: Linear elastic, Young's modulus = 3.0×10^6, Poisson's ratio = 0, density = 0.03.

Boundary conditions: u_x=u_y=0 at node 1, u_y=0 at node 3.

Loading: F_x=−500 at node 2, F_y=−1000 at node 2.

Truss section data: T2D2, cross-sectional area = 1.

Results and discussion

The SLIDERMPC keeps a node on a straight line between two nodes but allows it to slide along the line and the line to change length. This solution is obtained.

Input files

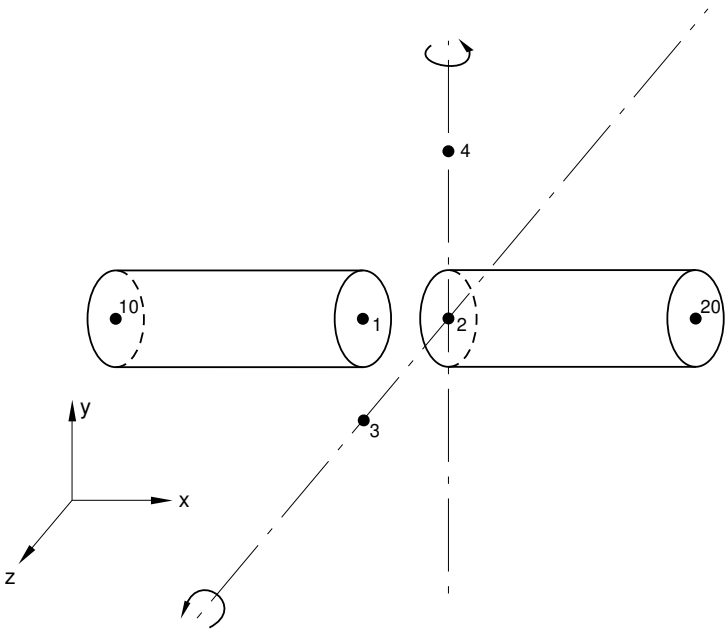
mpc_slider.inp SLIDERMPC.

UNIVERSALMPC

Elements tested

B33H

Problem description



PIN,1,2
UNIVERSAL,2,1,3,4

The UNIVERSALMPC is tested in Abaqus/Standard.

Boundary conditions: All degrees of freedom are restrained at node 10 throughout the analysis. Nodes 3 and 4 are initially constrained in degree of freedom 6.

Loading:

- Step 1: A concentrated follower force is applied at node 20 to pull the joint.
- Step 2: The joint is rotated by 45° about the 1–3 joint axis by prescribing degree of freedom 6 at node 3.

- Step 3: The joint is rotated by 45° about the current 1–4 axis by prescribing degree of freedom 6 at node 4.

Results and discussion

The axial follower force of Step 1 couples with the rotations in subsequent steps to cause lateral deflection of node 1 despite a very high material modulus.

Input files

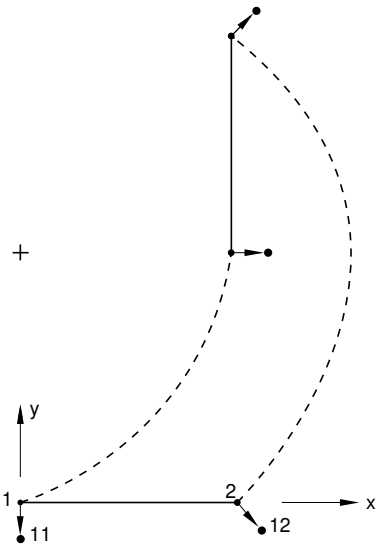
xmpcuniv.inp	UNIVERSAL and PINMPCs.
xmpcunivt.inp	UNIVERSAL and PINMPCs with transforms.

V LOCALMPC

Elements tested

B31H

Problem description



The V LOCALMPC is tested in Abaqus/Standard.

Boundary conditions:

ϕ_x=ϕ_y=0 at node 1, u_x=u_y=u_z=0 at node 11, and u_x=u_z=0 at node 12 in Steps 1 and 2.

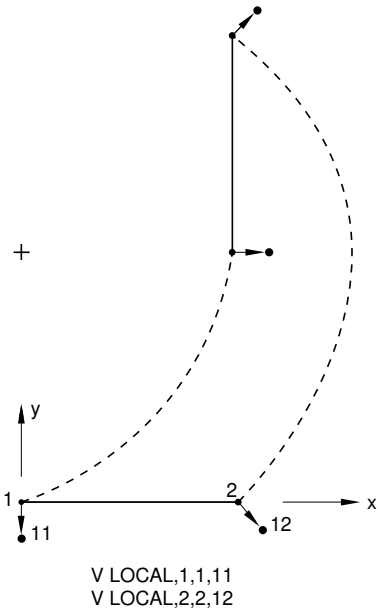
Loading:

- Step 1: Uniform load P=1.0 along the element.

- Step 2: The loads that were applied in the first step are applied again, this time using NLGEOM for large-displacement analysis.
- Step 3: Set u_y=15.708 at node 11 to push the beam.

Results and discussion

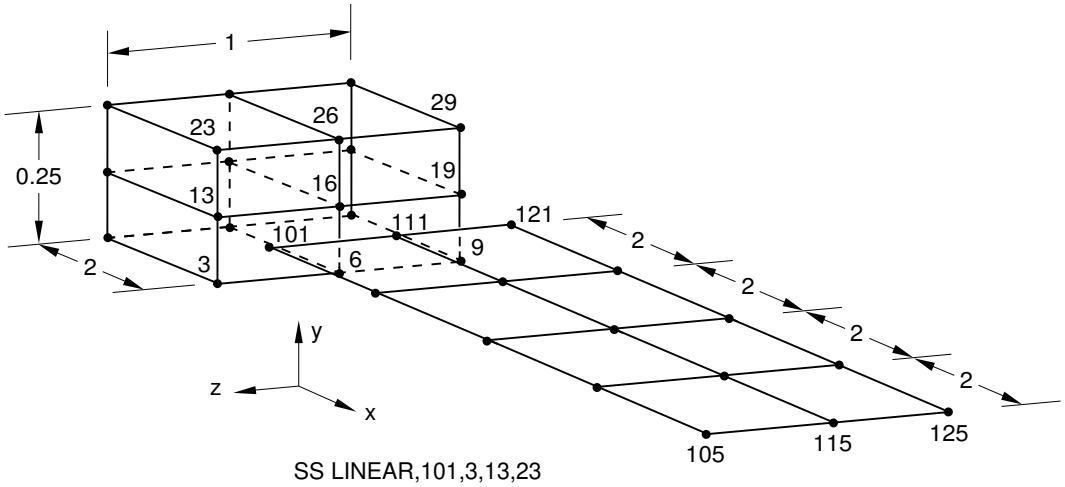
The constrained nodes move as predicted by the velocity constraint.



Input files

xmpcvloc.inp	V LOCALMPC.
xmpcvloct.inp	V LOCALMPC with transforms.

SS LINEAR and SLIDERMPCs




```
SS LINEAR,101,3,13,23
SS LINEAR,111,6,16,26
SS LINEAR,121,9,19,29
SLIDER,13,3,23
SLIDER,16,6,26
SLIDER,19,9,29
```

The SS LINEAR and SLIDERMPCs are tested in Abaqus/Standard and Abaqus/Explicit. A cantilever beam consisting of solid and shell elements connected by SS LINEAR and SLIDERMPCs is subjected to a transverse tip loading.

Initial Abaqus/Standard analysis

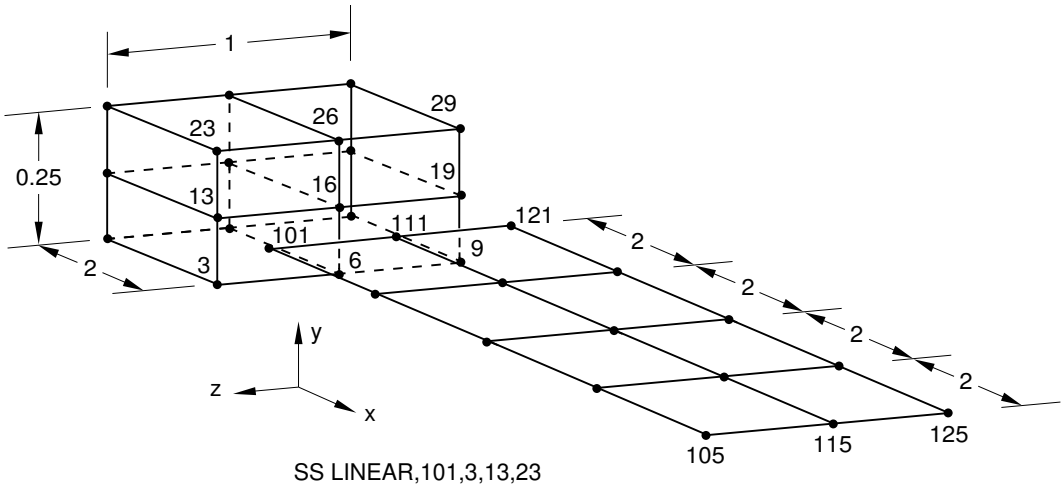
Elements tested

C3D8 S4R

Problem description

Loading:

- Step 1: F_y=−15 at nodes 105 and 125, F_y=−30 at node 115.



SS LINEAR,101,3,13,23
SS LINEAR,111,6,16,26
SS LINEAR,121,9,19,29
SLIDER,13,3,23
SLIDER,16,6,26
SLIDER,19,9,29

- Step 2: The loads that were applied in the first step are applied again, this time using NLGEOM for large-displacement analysis.
- Step 3: The loads that were applied in the second step are removed.
- Step 4: The boundary conditions are changed, and a rotation of $\pi/2$ around the z-axis is prescribed at x=0.

Initial boundary conditions: $u_x=u_y=u_z=0$ at $x=0$, $u_z=\phi_x=\phi_y=0$ at $z=0$ (except at nodes 19 and 121).

Boundary conditions in Step 4: $u_x=u_y=u_z=0$ and ϕ_z prescribed at x=10.

Results and discussion

The SLIDERMPC is used to keep a node on a straight line between two nodes, but it allows the node to slide along the line and the line to change length. This enforces the assumption that plane sections remain plane. The SS LINEARMPC constrains a shell node to a line of solid element nodes. This ties the translation and rotation of the shell node to the displacement and rotation of the solid nodes. Continuity of displacements and rotations is achieved at the shell-solid boundary.

Note:

The poor performance of the first-order brick element, C3D8, in bending is demonstrated by an excessively stiff response in Step 1 and Step 2.

Input files

xmpcssli.inp	SS LINEAR and SLIDERMPCs.
xmpcsslit.inp	SS LINEAR and SLIDERMPCs with transforms.

Abaqus/StandardRIKS analysis

Elements tested

C3D8 S4R

Problem description

Boundary conditions: $u_x=u_y=u_z=0$ at $x=0$, $u_z=\phi_x=\phi_y=0$ at $z=0$ (except at nodes 19 and 121).

Loading: $F_y=-15$ at nodes 105 and 125, $F_y=-30$ at node 115. A static Riks step is adopted.

Results and discussion

The SLIDERMPC is used to keep a node on a straight line between two nodes, but it allows the node to slide along the line and the line to change length. This enforces the assumption that plane sections remain plane. The SS LINEARMPC constrains a shell node to a line of solid element nodes. This ties the translation and rotation of the shell node to the displacement and rotation of the solid nodes. Continuity of displacements and rotations is achieved at the shell-solid boundary.

Input files

xmpcsslr.inp	SS LINEAR and SLIDERMPCs with RIKS.
xmpcsslrt.inp	SS LINEAR and SLIDERMPCs with RIKS and transforms.

Dynamic Abaqus/Standard analysis

Elements tested

C3D8 S4R

Problem description

Boundary conditions: The edge at $x=10$ is fixed.

Loading:

- Step 1: The first four natural frequencies are extracted.
- Step 2: $F_y=-30$ at all nodes along $x=0$. A large-displacement analysis is performed.
- Step 3: The load applied in Step 2 is removed. A dynamic analysis is performed.

Results and discussion

The SLIDERMPC is used to keep a node on a straight line between two nodes, but it allows the node to slide along the line and the line to change length. This enforces the assumption that plane sections remain plane. The SS LINEARMPC constrains a shell node to a line of solid element nodes. This ties the translation and rotation of the shell node to the displacement and rotation of the solid nodes. Continuity of displacements and rotations is achieved at the shell-solid boundary.

Input files

xmpcssld.inp	SS LINEAR and SLIDERMPCs with **DYNAMIC*.
xmpcssldt.inp	SS LINEAR and SLIDERMPCs with **DYNAMIC* and transforms.

Abaqus/Explicit analysis

Elements tested

C3D8R S4R

Problem description

Material: Linear elastic, Young's modulus = 30.0×10^6, Poisson's ratio = 0.3, density = 0.3.

Boundary conditions: $u_x = u_y = u_z =$ 0 at $x =$ 0, $u_z = \phi_x = \phi_y =$ 0 at $z =$ 0.

Loading: $F_y =$ −15 at nodes 105 and 125, $F_y =$ −30 at node 115.

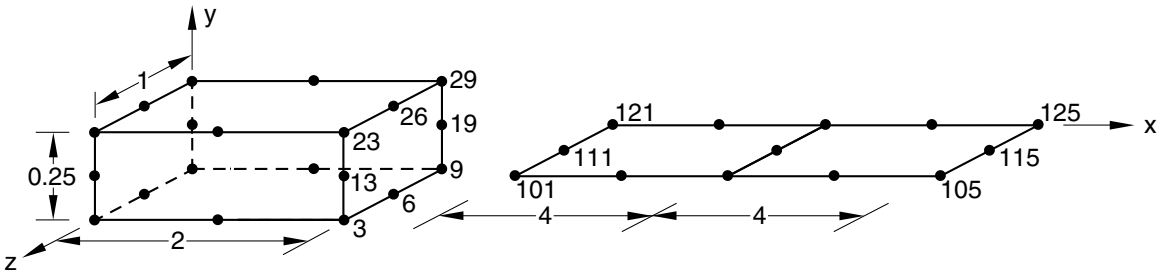
Results and discussion

The SLIDERMPC is used to keep a node on a straight line between two nodes, but it allows the node to slide along the line and the line to change length. This enforces the assumption that plane sections remain plane. The SS LINEARMPC constrains a shell node to a line of solid element nodes. This ties the translation and rotation of the shell node to the displacement and rotation of the solid nodes. Continuity of displacements and rotations is achieved at the shell-solid boundary.

Input files

mpc_sslinear.inp	SS LINEAR and SLIDERMPCs.

SS BILINEAR, SSF BILINEAR, and SLIDERMPCs




```
SS BILINEAR,101,3,13,23
SS BILINEAR,121,9,19,29
SSF BILINEAR,111,3,6,9,13,19,23,26,29
SLIDER,13,3,23
SLIDER,19,9,29
```

The SS BILINEAR, SSF BILINEAR, and SLIDERMPCs are tested in Abaqus/Standard.

Initial analysis

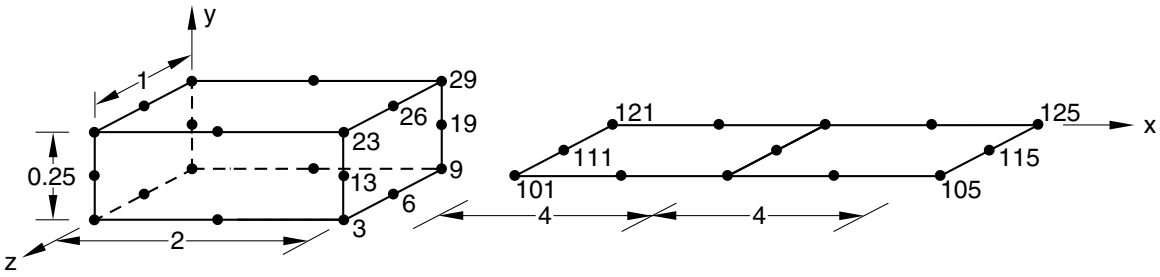
Elements tested

C3D20 S8R

Problem description

Loading:

- Step 1: F_y=−15 at nodes 105 and 125, F_y=−30 at node 115.



SS BILINEAR,101,3,13,23
SS BILINEAR,121,9,19,29
SSF BILINEAR,111,3,6,9,13,19,23,26,29
SLIDER,13,3,23
SLIDER,19,9,29

- Step 2: The loads that were applied in the first step are applied again, this time using NLGEOM for large-displacement analysis.
- Step 3: The loads that were applied in the second step are removed.
- Step 4: The boundary conditions are changed, and a rotation of $\pi/2$ around the z-axis is prescribed at x=0.

Initial boundary conditions: u_x=u_y=u_z=0 at x=0, u_z=ϕ_x=ϕ_y=0 at z=0 (except at nodes 19 and 121).

Boundary conditions in Step 4: u_x=u_y=u_z=0 and ϕ_z prescribed at x=10.

Results and discussion

Continuity of displacements and rotations is achieved at the shell-solid boundary.

Input files

xmpcssbi.inp	SS BILINEAR, SSF BILINEAR, and SLIDERMPCs.
xmpcssbit.inp	SS BILINEAR, SSF BILINEAR, and SLIDERMPCs with transforms.

RIKS analysis

Elements tested
C3D20 S8R

Problem description

Boundary conditions: $u_x=u_y=u_z=0$ at $x=0$, $u_z=\phi_x=\phi_y=0$ at $z=0$ (except at nodes 19 and 121).

Loading: $F_y=-15$ at nodes 105 and 125, $F_y=-30$ at node 115. A static Riks step is adopted.

Results and discussion

Continuity of displacements and rotations is achieved at the shell-solid boundary.

Input files

xmpcssbr.inp	SS LINEAR, SSF BILINEAR, and SLIDERMPCs with RIKS.
xmpcssbrt.inp	SS LINEAR, SSF BILINEAR, and SLIDERMPCs with RIKS and transforms.

Dynamic analysis

Elements tested
C3D20 S8R

Problem description

Boundary conditions: The edge at $x=10$ is fixed.

Loading:

- Step 1: The first four natural frequencies are extracted.
- Step 2: $F_y=-30$ at all nodes along $x=0$. A large-displacement analysis is performed.
- Step 3: The load applied in Step 2 is removed. A dynamic analysis is performed.

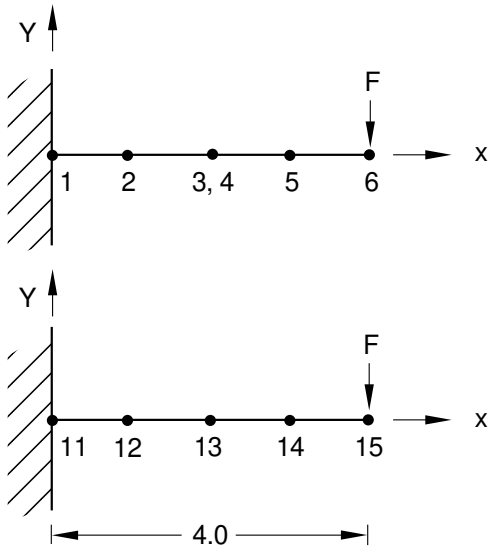
Results and discussion

Continuity of displacements and rotations is achieved at the shell-solid boundary.

Input files

xmpcssbd.inp	SS LINEAR, SSF BILINEAR, and SLIDERMPCs with **DYNAMIC*.

TIEMPC



TIE,4,3

The TIEMPC is tested in Abaqus/Standard and Abaqus/Explicit. A cantilevered beam is subjected to a transverse tip load.

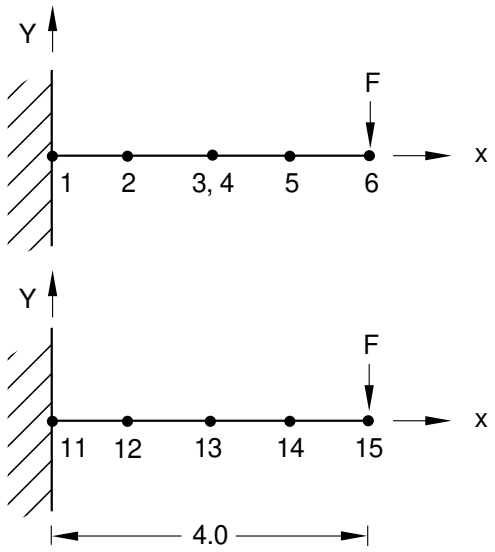
Initial Abaqus/Standard analysis

Elements tested

B22

Problem description

Material: Linear elastic, Young's modulus = 28.1×10^6, Poisson's ratio = 0.3, density = 1700.



TIE,4,3

Boundary conditions: Nodes 1 and 11 are clamped.

Loading:

- Step 1: F_y=−300 at nodes 6 and 15. A linear perturbation analysis is performed.
- Step 2: The natural frequencies and mode shapes for the continuous cantilever beam are extracted.
- Step 3: The natural frequencies and modes shapes are extracted for the cantilever beam that uses MPCTIE.
- Step 4: The loads that were applied in the first step are applied again, this time using NLGEOM for large-displacement analysis.

Results and discussion

MPCTIE makes all active degrees of freedom equal between two nodes (both translational and rotational degrees of freedom). The results of a cantilever beam that uses MPCTIE are the same as those of a continuous cantilever beam under the same loading.

Input files

xmpctiex.inp	TIEMPC.
xmpctiext.inp	TIEMPC with transforms.

Abaqus/StandardRIKS analysis

Elements tested

B22

Problem description

A cantilever beam with MPC type TIE, subject to a slight imperfection corresponding to the first buckling mode.

Material: Linear elastic, Young's modulus = 28.1×10^6, Poisson's ratio = 0.3, density = 1700.

Boundary conditions: Node 1 is clamped.

Loading:

- Step 1: The first four buckling modes are extracted for a perturbation load F_x=−300 at node 6.
- Step 2: A RIKS analysis (with NLGEOM) is conducted until a maximum load of F_x=−600 at node 6.

Results and discussion

MPCTIE makes all active degrees of freedom equal between two nodes (both translational and rotational degrees of freedom). The results of a cantilever beam that uses MPCTIE are the same as those of a continuous cantilever beam under the same loading.

Input files

xmpctier.inp	TIEMPC with RIKS.
xmpctiert.inp	TIEMPC with RIKS and transforms.

Abaqus/Explicit analysis

Elements tested

B21 PIPE21

Problem description

The following equivalent cases are considered:

1. A TIE-type MPC is defined between nodes 3 and 4.
2. Nodes 3 and 4 are included in a rigid body tie-type node set.

The results from the above two cases are compared to the solution of a continuous cantilever beam under the same transverse tip loading.

Material: Linear elastic, Young's modulus = 28.1×10^6, Poisson's ratio = 0.3, density = 0.3.

Boundary conditions: Nodes 1 and 11 are clamped.

Loading: $F_y = -300$ at nodes 6 and 15.

Beam section data: B21, 0.5×0.5 rectangle.

PIPE21, pipe with radius 0.5 and thickness 0.05.

Results and discussion

MPCTIE makes all active degrees of freedom equal between two nodes (both translational and rotational degrees of freedom). The results of a cantilever beam that uses MPCTIE are the same as those of a continuous cantilever beam under the same loading. The results from Case 2 match the results from Case 1.

Input files

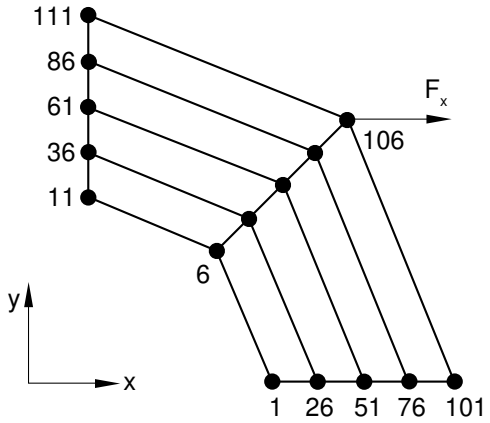
mpc_tie.inp	Input data for Case 1 for beam elements.
mpc_tierig.inp	Input data for Case 2 for beam elements.
mpc_tie_pipe.inp	Input data for Case 1 for pipe elements.
mpc_tierig_pipe.inp	Input data for Case 2 for pipe elements.

CYCLSYMMPC

Elements tested

CPE4 CPE4T CPEG4T

Problem description




```
CYCLSYM,1,11
CYCLSYM,26,36
CYCLSYM,51,61
CYCLSYM,76,86
CYCLSYM,101,111
```

The CYCLSYMMPC is tested in Abaqus/Standard. A disk is subjected to cyclic symmetric force loading in the first analysis; in the second analysis the disk is subjected to both cyclic symmetric force loading and cyclic temperature boundary conditions. The problem is modeled using a quarter of the disk with the appropriate CYCLSYMMPC.

Boundary conditions: Nodes 6 and 11 are clamped. The reference node for the CPEG4T model is also clamped. Node 1 also has all displacement and rotation degrees of freedom restrained because of the CYCLSYMMPC. Nodes 6, 11, and 1 have their temperatures set to zero for the second analysis.

Loading: F_x=100 at node 106. For the second analysis the temperature of nodes 101 and 111 is set to 100, and the temperature of node 106 is set to 200.

The first analysis uses the direct-integration implicit dynamic procedure; the second analysis uses the fully coupled thermal-stress steady-state procedure.

Results and discussion

The results obtained from the quarter disk model that uses MPC type CYCLSYM are the same as the results obtained from an analysis of a complete disk under cyclic symmetric loading and subjected to cyclic temperature boundary conditions.

Input files

xmpccycd.inp	CYCLSYMMPC with **DYNAMIC*.
xmpccyct.inp	CYCLSYMMPC with **COUPLED TEMPERATURE-DISPLACEMENT*.
xmpccyct_cpeg4t.inp	CYCLSYMMPC with **COUPLED TEMPERATURE-DISPLACEMENT*.

Internal MPC types BEAMRIGID and BEAMTIE with transforms

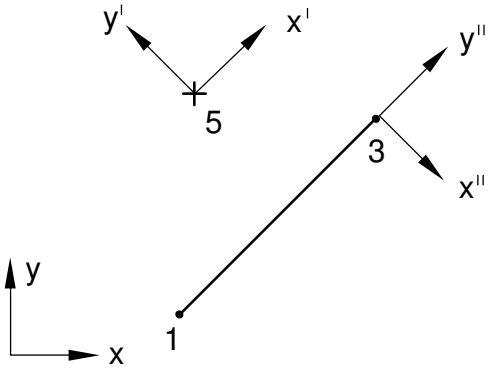
These files test the use of the internally generated MPCs (MPC types BEAMRIGID and BEAMTIE) with transforms in Abaqus/Standard. Transformations are applied to the reference node as well as to the nodes of the rigid element (or rigid beam). The boundary conditions and loadings, mentioned below, are given in the local transformed system.

Rigid elements

Elements tested

R2D2 R3D4

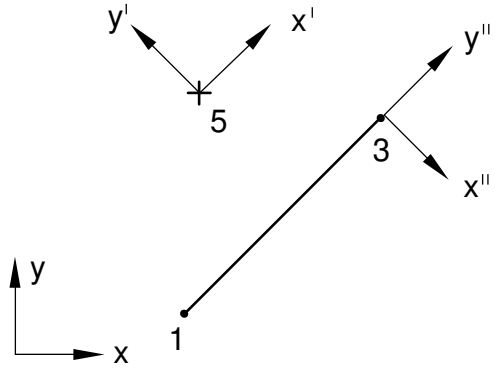
Problem description



Boundary conditions: u_x=0 and u_y=1.5 at node 5.

Loading:

- Step 1: F_y=10.0 at node 3.
- Step 2: Same as above, but a large-displacement analysis is performed.



Results and discussion

The results agree with the theoretically expected results. The results of the geometrically nonlinear analyses show that the initial stress terms are accounted for correctly.

Input files

xmpcrgd2.inp	R2D2 elements.
xmpcrgd3.inp	R3D4 elements.

Rigid beams

Elements tested

RB2D2 RB3D2

Problem description

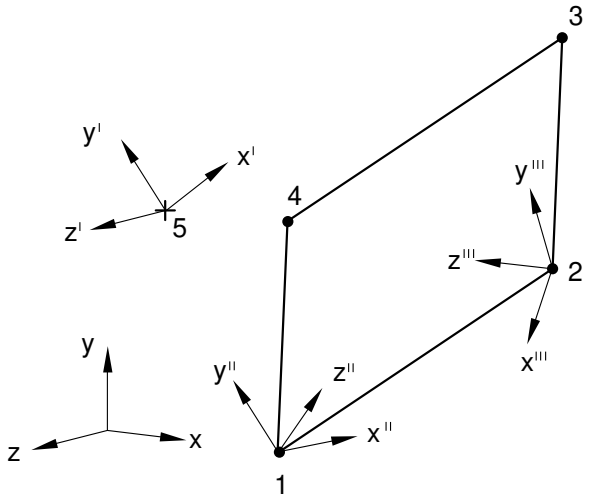
Boundary conditions: u_y=1.5 at node 5. All other displacements are fixed.

Loading:

- Step 1: F_z=10.0 at node 1.
- Step 2: Same as above, but a large-displacement analysis is performed.

Results and discussion

The results agree with the theoretically expected results. The results of the geometrically nonlinear analyses show that the initial stress terms are accounted for correctly.



Input files

xmpcrgb2.inp	RB2D2 elements.
xmpcrgb3.inp	RB3D2 elements.

MPC sorting

Elements tested

S4R

Problem description

MPC sorting is tested in Abaqus/Standard.The model is a cantilever structure composed of 20 shell elements tied together using MPC type TIE.

Boundary conditions: One end of the structure is clamped.

Loading: A concentrated load of F_y=1.0 is applied at the other end of the structure.

Results and discussion

Abaqus successfully sorts the MPC definitions such that no input errors occur.

Input files

xmpcsort.inp	Test of internal sorting of MPC type TIE.

Surface-based tie constraint

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

AC2D3 AC2D4 AC2D4R AC2D6 AC2D8
AC3D4 AC3D5 AC3D6 AC3D8 AC3D8R
AC3D10 AC3D15 AC3D20
ACAX3 ACAX4 ACAX4R ACAX6 ACAX8
C3D4 C3D8 C3D8R C3D10 C3D10M
C3D4T C3D6T C3D8RT C3D8P C3D8T C3D8PT C3D10MP
C3D15 C3D20 C3D20E C3D20P
CAX4 CAX4R CAX4T CAX8 CAX3T CAX8T
CPE3 CPE4R CPE3T CPE4 CPE4P CPE6 CPE6MP CPE8 CPE8P
CPS3T CPS4 CPS4R CPS8
DC3D8 DC3D8E DCAX4 DS4 DSAX1 DSAX2
MAX2
R2D2 R3D4
B21 B31 PIPE21 PIPE31
M3D4R
S3RT S4 S4R S4RT S4T S8RT SAX1 SAX2 SAX2T SC6RT SC8RT

Features tested

The surface-based tie constraint is tested for a number of general cases and for the special case of acoustic-structural coupling.

Problem description

These tests verify the performance of the surface-based tie constraint for various analyses using acoustic, continuum, and shell elements with the surfaces defined in different ways.

Results and discussion

The results for the general cases indicate that the surfaces can be adjusted and tied appropriately.

In the suite of coupled acoustic-structural input files each of the acoustic element types is tested in both secondary and main roles, tied to main surfaces formed of solid continuum elements of similar interpolation order. In addition, the suite includes input files testing the quadratic acoustic element types in the secondary role, with linear solid continuum elements forming the main surfaces. The results indicate that the fluid-solid coupling functions correctly.

Input files

Abaqus/Standard input files

xtie_solid.inp	Tie solid elements with surfaces defined in different ways.
xtie_solid_combine.inp	Tie solid elements with surfaces combined.
xtie_shell.inp	Tie shell elements with surfaces defined in different ways.
xtie_shell_norot.inp	Tie shell elements with surfaces defined in different ways. The rotational degrees of freedom are not tied.
xtie_shell_beam.inp	Tie beam and shell elements together.
xtie_solid_shell.inp	Tie solid and shell elements together.
xtie_shell_shell.inp	Tie two shell elements that are perpendicular to each other.
xtie_shell_shell_norot.inp	Tie two shell elements that are perpendicular to each other. The rotational degrees of freedom are not tied.
xtie_shell_shell_nothick.inp	Tie two shell elements that are perpendicular to each other without accounting for the shell thickness.
xtie_shell_shell_offset.inp	Tie shell elements defined using shell offset.
xtie_c3d20_c3d8.inp	Tie second-order and first-order solid elements together.
xtie_cax4_sax1.inp	Tie first-order axisymmetric solid and shell elements together.
xtie_cax8_sax2.inp	Tie second-order axisymmetric solid and shell elements together.
xtie_r2d2.inp	Tie two-dimensional solid elements to a surface defined by a rigid element.
xtie_r3d4.inp	Tie three-dimensional solid elements to a surface defined by a rigid element.
xtie_rigid2d.inp	Tie two-dimensional solid elements to an analytical rigid surface.
xtie_rigid3d.inp	Tie three-dimensional solid elements to an analytical rigid surface.
xtie_elec_heat.inp	Tie elements in a coupled electrical-heat analysis.
xtie_shell_axisy_heat.inp	Tie two axisymmetric shell elements in a heat transfer analysis.
xtie_shell_heat.inp	Tie two shell elements in a heat transfer analysis.
xtie_shell_solid_axisy_heat.inp	Tie axisymmetric shell and solid elements together in a heat transfer analysis.
xtie_shell_solid_heat.inp	Tie shell and solid elements together in a heat transfer analysis.
xtie_shell_axisy_couple.inp	Tie axisymmetric shell elements in a coupled thermal-stress analysis.
xtie_shell_couple.inp	Tie two shell elements in a coupled thermal-stress analysis.
xtie_shell_solid_axisy_couple.inp	Tie axisymmetric shell and solid elements together in a coupled thermal-stress analysis.
xtie_shell_solid_couple.inp	Tie shell and solid elements together in a coupled thermal-stress analysis.
xtie_s4t.inp	Tie S4T elements together in a coupled thermal-stress analysis.
xtie_s4rt.inp	Tie S4RT elements together in a coupled thermal-stress analysis.
xtie_s3rt.inp	Tie S3RT elements together in a coupled thermal-stress analysis.
xtie_cpe3t.inp	Tie CPE3T elements together in a coupled thermal-stress analysis.
xtie_cps3t.inp	Tie CPS3T elements together in a coupled thermal-stress analysis.
xtie_cax3t.inp	Tie CAX3T elements together in a coupled thermal-stress analysis.
xtie_c3d4t.inp	Tie C3D4T elements together in a coupled thermal-stress analysis.
xtie_c3d6t.inp	Tie C3D6T elements together in a coupled thermal-stress analysis.

xtie_sc6rt.inp	Tie SC6RT elements together in a coupled thermal-stress analysis.
xtie_sc8rt.inp	Tie SC8RT elements together in a coupled thermal-stress analysis.
xtie_cpe4p.inp	Tie CPE4P elements with the surfaces defined in different ways.
xtie_cpe8p.inp	Tie CPE8P elements with the surfaces defined in different ways.
xtie_cpe6mp.inp	Tie CPE6MP elements with the surfaces defined in different ways.
xtie_c3d8p.inp	Tie C3D8P elements with the surfaces defined in different ways.
xtie_c3d10mp.inp	Tie C3D10MP elements with the surfaces defined in different ways.
xtie_c3d20p.inp	Tie C3D20P elements with the surfaces defined in different ways.
xtie_piezo.inp	Tie continuum and piezoelectric elements.
xtie_rigid_couple.inp	Tie a three-dimensional element to a rigid surface in a coupled thermal-stress analysis.
xtie_analyt_rigid_couple.inp	Tie a three-dimensional element to an analytical rigid surface in a coupled thermal-stress analysis.
xtie_solid_2d_2ties.inp	Use two pairs of tie constraints in two dimensions.
xtie_solid_3d_5ties.inp	Use five pairs of tie constraints in three dimensions.
xtie_cax8t_sax2t_cax8t.inp	Tie a SAX2T element with two CAX8T elements in a coupled thermal-stress analysis.
xtie_cax8_max2_cax8.inp	Tie a MAX2 element with two CAX8 elements.
xtie_isolated_nodes.inp	Tie two-dimensional elements to isolated nodes.
ec234afat.inp	AC2D3 and CPS4 elements in a coupled acoustic-solid analysis.
ec244afat.inp	AC2D4 and CPS4 elements in a coupled acoustic-solid analysis.
ec264afat.inp	AC2D6 and CPS4 elements in a coupled acoustic-solid analysis (acoustic secondary-only case).
ec268afat.inp	AC2D6 and CPS8 elements in a coupled acoustic-solid analysis.
ec284afat.inp	AC2D8 and CPS4 elements in a coupled acoustic-solid analysis (acoustic secondary-only case).
ec288afat.inp	AC2D8 and CPS8 elements in a coupled acoustic-solid analysis.
ec348afat.inp	AC3D4 and C3D8 elements in a coupled acoustic-solid analysis.
ec358afat.inp	AC3D5 and C3D8 elements in a coupled acoustic-solid analysis.
ec368afat.inp	AC3D6 and C3D8 elements in a coupled acoustic-solid analysis.
ec388afat.inp	AC3D8 and C3D8 elements in a coupled acoustic-solid analysis.
ec3a8afat.inp	AC3D10 and C3D8 elements in a coupled acoustic-solid analysis (acoustic secondary-only case).
ec3afafat.inp	AC3D10 and C3D15 elements in a coupled acoustic-solid analysis.
ec3f8afat.inp	AC3D15 and C3D8 elements in a coupled acoustic-solid analysis (acoustic secondary-only case).
ec3ffafat.inp	AC3D15 and C3D15 elements in a coupled acoustic-solid analysis.
ec3k8afat.inp	AC3D20 and C3D8 elements in a coupled acoustic-solid analysis (acoustic secondary-only case).
ec3kkafat.inp	AC3D20 and C3D20 elements in a coupled acoustic-solid analysis.
eca34afat.inp	ACAX3 and CAX4 elements in a coupled acoustic-solid analysis.
eca44afat.inp	ACAX4 and CAX4 elements in a coupled acoustic-solid analysis.
eca64afat.inp	ACAX6 and CAX4 elements in a coupled acoustic-solid analysis (acoustic secondary-only case).
eca68afat.inp	ACAX6 and CAX8 elements in a coupled acoustic-solid analysis.

eca84afat.inp	ACAX8 and CAX4 elements in a coupled acoustic-solid analysis (acoustic secondary-only case).
eca88afat.inp	ACAX8 and CAX8 elements in a coupled acoustic-solid analysis.
tie_moddyn_ac2d3.inp	AC2D3 and CPS4R elements in a coupled acoustic-solid analysis using the **DYNAMIC* and **MODAL DYNAMIC* analysis types.
tie_moddyn_ac2d4.inp	AC2D4 and CPS4R elements in a coupled acoustic-solid analysis using the **DYNAMIC* and **MODAL DYNAMIC* analysis types.
tie_moddyn_ac3d4.inp	AC3D4 and C3D8R elements in a coupled acoustic-solid analysis using the **DYNAMIC* and **MODAL DYNAMIC* analysis types.
tie_moddyn_ac3d5.inp	AC3D5 and C3D8R elements in a coupled acoustic-solid analysis using the **DYNAMIC* and **MODAL DYNAMIC* analysis types.
tie_moddyn_ac3d6.inp	AC3D6 and C3D8R elements in a coupled acoustic-solid analysis using the **DYNAMIC* and **MODAL DYNAMIC* analysis types.
tie_moddyn_ac3d8.inp	AC3D8 and C3D8R elements in a coupled acoustic-solid analysis using the **DYNAMIC* and **MODAL DYNAMIC* analysis types.
tie_moddyn_acax3.inp	ACAX3 and CAX4R elements in a coupled acoustic-solid analysis using the **DYNAMIC* and **MODAL DYNAMIC* analysis types.
tie_moddyn_acax4.inp	ACAX4 and CAX4R elements in a coupled acoustic-solid analysis using the **DYNAMIC* and **MODAL DYNAMIC* analysis types.
tie_shell_cgax4.inp	Test tie between SAX1 element and CGAX4 element for rotation when the secondary and main surfaces are apart.
tie_beam_cps4.inp	Test tie between B21 element and CPS4 element for rotation when the secondary and main surfaces are apart.
tie_beam_memb.inp	Test tie between B31 element and M3D4 element for rotation when the secondary and main surfaces are apart.
tie_beam_surf.inp	Test tie between B31 element and SFM3D4 element for rotation when the secondary and main surfaces are apart.
tie_cps4_beam.inp	Test tie between CPS4 element and B21 element when CPS4 is the main and the secondary and main surfaces are apart.
tie_cax4_shell.inp	Test tie between SAX1 element and CAX4 element when CAX4 is the main and the secondary and main surfaces are apart.
tie_c3d8_shell.inp	Test tie between C3D8 element and S4R element when C3D8 is the main and the secondary and main surfaces are apart.
tie_shell_shell_constraint.inp	Test tie between S4R element and S4R element when the CONSTRAINT RATIO parameter is specified.
tie_acoinf_edge.inp	Test edge-to-edge tie for ACIN3D4 elements.
tie_memb_memb_edge.inp	Test edge-to-edge tie for M3D4 elements.
tie_memb_rigid_edge.inp	Test edge-to-edge tie between M3D4 and R3D4 elements.
tie_shell_shell_edge.inp	Test edge-to-edge tie for S8R elements.
tie_surf_surf_edge.inp	Test edge-to-edge tie for SFM3D4 elements.
ctp_tie.inp	Test surface tie for C3D8PT elements.
ctp_tie_nodesurf.inp	Test nodal surface tie for C3D8PT elements.

Abaqus/Explicit input files

xtie_xpl_solid.inp	Tie solid elements with surfaces defined in different ways.
xtie_xpl_c3d8.inp	Tie C3D8 elements with surfaces defined in different ways.

xtie_xpl_sc8rt_couple.inp	Tie SC8RT elements with surfaces defined in different ways.
xtie_xpl_solid_ss.inp	Same as xtie_xpl_solid.inp except surface-to-surface tie formulation is used.
xtie_xpl_shell.inp	Tie shell elements with surfaces defined in different ways.
xtie_xpl_shell_s4.inp	Tie S4 elements with surfaces defined in different ways.
xtie_xpl_shell_ss.inp	Same as xtie_xpl_shell.inp except surface-to-surface tie formulation is used.
xtie_xpl_shell_norot.inp	Tie shell elements with surfaces defined in different ways. The rotational degrees of freedom are not tied.
xtie_xpl_solid_shell.inp	Tie solid and shell elements together.
xtie_xpl_solid_shell_ss.inp	Same as xtie_xpl_solid_shell.inp except surface-to-surface tie formulation is used.
xtie_xpl_shell_shell.inp	Tie two shell elements that are perpendicular to each other.
xtie_xpl_shell_shell_norot.inp	Tie two shell elements that are perpendicular to each other. The rotational degrees of freedom are not tied.
xtie_xpl_shell_shell_nothick.inp	Same as xtie_xpl_shell_shell.inp except thickness is not considered in the tie constraints.
xtie_xpl_shell_shell_ss_nothick.inp	Same as xtie_xpl_shell_shell.inp except surface-to-surface tie formulation is used and thickness is not considered in the tie constraints.
xtie_xpl_shell_cratio.inp	Tie two shell elements that are not adjusted. The rotational degrees of freedom are not tied and a constraint ratio value is prescribed.
xtie_xpl_cax4_sax1.inp	Tie first-order axisymmetric solid and shell elements together.
xtie_xpl_beam.inp	Tie beam elements together.
xtie_xpl_beamsolid.inp	Tie beam and solid elements together.
xtie_xpl_beamshell.inp	Tie beam and shell elements together.
xtie_xpl_beammembrane.inp	Tie beam and membrane elements together.
xtie_xpl_pipe.inp	Tie pipe elements together.
xtie_xpl_pipesolid.inp	Tie pipe and solid elements together.
xtie_xpl_pipeshell.inp	Tie pipe and shell elements together.
xtie_xpl_pipemembrane.inp	Tie pipe and membrane elements together.
xtie_xpl_r2d2.inp	Tie two-dimensional solid elements to a surface defined by a rigid element.
xtie_xpl_r3d4.inp	Tie three-dimensional solid elements to a surface defined by a rigid element.
xtie_xpl_rigid2d.inp	Tie two-dimensional solid elements to an analytical rigid surface.
xtie_xpl_rigid3d.inp	Tie three-dimensional solid elements to an analytical rigid surface.
xtie_xpl_rigrig2d.inp	Tie two-dimensional rigid bodies.
xtie_xpl_rigrig3d.inp	Tie three-dimensional rigid bodies.
xtie_xpl_solid_couple.inp	Tie three-dimensional solid elements in a coupled thermal-stress analysis.
xtie_xpl_rigid_couple.inp	Tie a three-dimensional element to a rigid surface in a coupled thermal-stress analysis.
xtie_xpl_analyt_rigid_couple.inp	Tie a three-dimensional element to an analytical rigid surface in a coupled thermal-stress analysis.
xtie_xpl_solid_2d_2ties.inp	Use two pairs of tie constraints in two dimensions.

xtie_xpl_solid_2d_2ties_ss.inp	Same as xtie_xpl_solid_2d_2ties.inp except surface-to-surface tie formulation is used.
xtie_xpl_solid_3d_5ties.inp	Use five pairs of tie constraints in three dimensions.
xtie_xpl_solid_3d_5ties_ss.inp	Same as xtie_xpl_solid_3d_5ties.inp except surface-to-surface tie formulation is used.
xtie_xpl_shell_3d_4ties.inp	Use four pairs of tie constraints to tie five shells in three dimensions.
xtie_xpl_isolated_nodes.inp	Tie two-dimensional elements to isolated nodes.
xtie_xpl_ac2d3.inp	AC2D3 and CPS4R elements in a coupled acoustic-solid analysis.
xtie_xpl_ac2d4.inp	AC2D4R and CPS4R elements in a coupled acoustic-solid analysis.
xtie_xpl_ac3d4.inp	AC3D4 and C3D8R elements in a coupled acoustic-solid analysis.
xtie_xpl_ac3d6.inp	AC3D6 and C3D8R elements in a coupled acoustic-solid analysis.
xtie_xpl_ac3d8.inp	AC3D8R and C3D8R elements in a coupled acoustic-solid analysis.
xtie_xpl_acax3.inp	ACAX3 and CAX4R elements in a coupled acoustic-solid analysis.
xtie_xpl_acax4.inp	ACAX4R and CAX4R elements in a coupled acoustic-solid analysis.

Rigid bodies

This section defines the basic tests that are used to provide evidence that the implementation of rigid bodies produces the expected results.

In this section:

- *Rigid body mass properties*
- *Tie and pin node sets*
- *Rigid body as an MPC*
- *Rigid body constraint*
- *Including deformable element types in a rigid body*

Rigid body mass properties

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

B21 B21H B22 B31 B31H B31OS B31OSH B32 B32H
CAX3 CAX3H CAX4 CAX4H CAX4I CAX4IH CAX4R CAX6 CAX6M CAX6MH CAX8 CAX8H
CPE3 CPE3H CPE4 CPE4H CPE4I CPE4IH CPE4R CPE4RH CPE6 CPE6M CPE8 CPS3 CPS4 CPS4I CPS4R CPS6 CPS6M CPS8
C3D6 C3D6H C3D8 C3D8H C3D8R C3D8RH C3D10 C3D10M C3D15 C3D15H C3D15V C3D20 C3D20H C3D20R
FRAME2D FRAME3D
M3D3 M3D4 M3D4R M3D6 M3D8 MAX1 MAX2
PIPE21 PIPE31 PIPE31H
R2D2 R3D4 R3D3 RAX2
S3R S4 S4R S8R SAX1 SAX2
T2D2 T2D3 T2D3H T3D2 T3D3 T3D3H

Features tested

Computation of rigid body mass properties, repositioning of the rigid body reference node at the center of mass of the rigid body.

Problem description

This suite of problems tests the mass property computations of rigid bodies consisting of continuum and structural elements in Abaqus/Standard analyses and continuum, structural, and rigid elements in Abaqus/Explicit analyses. Five different rigid body geometry cases are considered:

1. A two-dimensional planar rigid body consisting of beam, continuum, and truss elements (and rigid elements in Abaqus/Explicit analyses).
2. A three-dimensional rigid body consisting of beam, continuum, and truss elements (and rigid elements in Abaqus/Explicit analyses).
3. A three-dimensional rigid body consisting of beam, membrane, shell, and truss elements.
4. An axisymmetric rigid body consisting of continuum and shell elements (and rigid elements in Abaqus/Explicit analyses).
5. A three-dimensional rigid body consisting of all of the elements included in geometry Cases 2 and 3, as well as a point mass element located at the rigid body reference node.

The mass, center of mass, and rotary inertia of each rigid body are computed automatically by Abaqus to take into account the section properties and densities of each of the constituent elements. The reference node for each rigid body is located at the center of mass.

The computed mass properties of rigid bodies can be verified by checking the printed quantities in the data (`.dat`) file. Further quantitative and qualitative verification is accomplished by performing two analyses. In the first analysis each geometry case is subjected to a concentrated force of magnitude 1.0×10^6 in the x-direction acting at the rigid body reference node. In the second analysis each geometry case is subjected to a concentrated moment of magnitude 1.0×10^8 acting about the z-axis at the rigid body reference node.

Results and discussion

For each geometry case the mass and inertia properties of the rigid body are found to match their analytical values closely. In Cases 1 and 4 the application of a concentrated force at the rigid body reference node does not cause any rotation of the rigid body about the out-of-plane axis, which verifies that the reference node has been positioned at the center of mass of the rigid body. Similarly for Cases 2, 3, and 5, for the concentrated force loading, there are no rotations observed about either the global x-, y-, or z-axes. The moment loading in each case causes large rigid body rotations about the reference node. The final rotated configuration in each case is found to be consistent with the geometry of the problem and the magnitude of the applied moment. The original and final configurations of the rigid body in Case 1 for the moment load case are shown in *Figure 1* and *Figure 2*.

Input files

Abaqus/Standard analyses

rigmass1_std.inp	Case 1 for the force loading.
rigmass1a_std.inp	Case 1 for the force loading.
rigmass1b_std.inp	Case 1 for the force loading.
rigmass1c_std.inp	Case 1 for the force loading.
rigmass11_std.inp	Case 1 for the moment loading.
rigmass11a_std.inp	Case 1 for the moment loading.
rigmass2_std.inp	Case 2 for the force loading.
rigmass2a_std.inp	Case 2 for the force loading.
rigmass2b_std.inp	Case 2 for the force loading.
rigmass2c_std.inp	Case 2 for the force loading.
rigmass2d_std.inp	Case 2 for the force loading.
rigmass2e_std.inp	Case 2 for the force loading.
rigmass2f_std.inp	Case 2 for the force loading.
rigmass22_std.inp	Case 2 for the moment loading.
rigmass22a_std.inp	Case 2 for the moment loading.
rigmass3_std.inp	Case 3 for the force loading.
rigmass3a_std.inp	Case 3 for the force loading.
rigmass3b_std.inp	Case 3 for the force loading.
rigmass33_std.inp	Case 3 for the moment loading.
rigmass33a_std.inp	Case 3 for the moment loading.
rigmass4_std.inp	Case 4 for the force loading.
rigmass4a_std.inp	Case 4 for the force loading.
rigmass4b_std.inp	Case 4 for the force loading.
rigmass4c_std.inp	Case 4 for the force loading.
rigmass4d_std.inp	Case 4 for the force loading.

rigmass44_std.inp	Case 4 for the moment loading.
rigmass44a_std.inp	Case 4 for the moment loading.
rigmass5_std.inp	Case 5 for the force loading.
rigmass55_std.inp	Case 5 for the moment loading.

Abaqus/Explicit analyses

rigmass1.inp	Case 1 for the force loading on linear elements.
rigmass1a.inp	Case 1 for the force loading on quadratic elements.
rigmass1_pipe.inp	Case 1 for the force loading on linear elements including pipe elements.
rigmass11.inp	Case 1 for the moment loading on linear elements.
rigmass11a.inp	Case 1 for the moment loading on quadratic elements.
rigmass11_pipe.inp	Case 1 for the moment loading on linear elements including pipe elements.
rigmass2.inp	Case 2 for the force loading on linear elements.
rigmass2a.inp	Case 2 for the force loading on quadratic elements.
rigmass2_pipe.inp	Case 2 for the force loading on linear elements including pipe elements.
rigmass2b.inp	Case 2 for the force loading on elements without reduced integration.
rigmass22.inp	Case 2 for the moment loading on linear elements.
rigmass22a.inp	Case 2 for the moment loading on quadratic elements.
rigmass22_pipe.inp	Case 2 for the moment loading on linear elements including pipe elements.
rigmass22b.inp	Case 2 for the moment loading on elements without reduced integration.
rigmass3.inp	Case 3 for the force loading on linear elements.
rigmass3a.inp	Case 3 for the force loading on quadratic beam elements.
rigmass3b.inp	Case 3 for the force loading on elements without reduced integration.
rigmass33.inp	Case 3 for the moment loading on linear elements.
rigmass33a.inp	Case 3 for the moment loading on quadratic beam elements.
rigmass33b.inp	Case 3 for the moment loading on elements without reduced integration.
rigmass4.inp	Case 4 for the force loading on linear elements.
rigmass4a.inp	Case 4 for the force loading on elements without reduced integration.
rigmass44.inp	Case 4 for the moment loading on linear elements.
rigmass44a.inp	Case 4 for the moment loading on elements without reduced integration.
rigmass5.inp	Case 5 for the force loading on linear elements.
rigmass5a.inp	Case 5 for the force loading on quadratic elements.
rigmass5b.inp	Case 5 for the force loading on elements without reduced integration.
rigmass55.inp	Case 5 for the moment loading on linear elements.
rigmass55a.inp	Case 5 for the moment loading on quadratic elements.
rigmass55b.inp	Case 5 for the moment loading on elements without reduced integration.

Figures

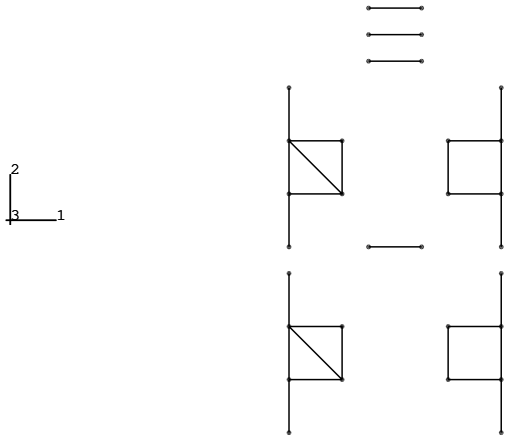


Figure 1: Original configuration for Case 1.

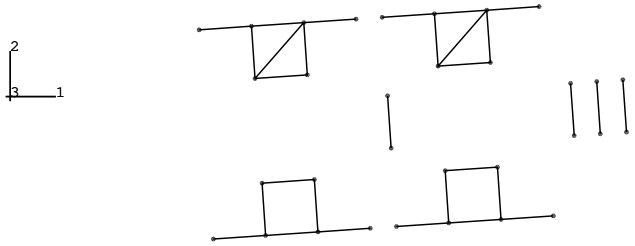


Figure 2: Final configuration for Case 1 subjected to applied torque about reference node.

Tie and pin node sets

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

R3D4 S4R

Features tested

Use of TIE NSET and PIN NSET to define connections between rigid bodies and deformable elements.

Problem description

A square rigid sheet is connected by one node at each of two opposite edges to deformable rectangular plates consisting of S4R elements. The connection to the first plate at node 6 of the rigid body is assumed to be a tie connection where it is desired to transmit moment and rotation. The connection to the second plate at node 8 is assumed to be a pin connection. A moment of magnitude 1000 is applied to the rigid body reference node about the global z-axis. A ROTARYI element with $\mathbf{I}_{zz} = 10$ is attached to the rigid body reference node. Two representations for the square rigid sheet are considered:

1. The rigid sheet is modeled with R3D4 elements. These elements have only translational degrees of freedom and, therefore, generate pin nodes on the rigid body by default. To ensure that there is a tie connection at node 6, the TIE NSET parameter is used with a node set containing node 6. For this model the PIN NSET parameter is also used with a node set containing node 8. However, this PIN NSET specification is not necessary (redundant) in this case since node 8 is by default a pin node because of the underlying R3D4 elements.
2. The rigid sheet is modeled with S4R elements. These elements have both translational and rotational degrees of freedom and, therefore, generate tie nodes on the rigid body by default. To ensure that there is a pin connection at node 8, the PIN NSET parameter is used with a node set containing node 8. For this model the TIE NSET parameter is also used with a node set containing node 6. However, this TIE NSET specification is not necessary in this case since node 6 is, by default, a tie node because of the underlying S4R elements.

Results and discussion

The original and final configurations for Cases 1 and 2 are shown in *Figure 1* and *Figure 2*. It is clear from the results that at tie connections the plate rotates with the rigid body since there is transfer of moment from the rigid sheet to the rectangular plate at the connecting node. At pin connections moments are not transferred at the connecting node since the rigid body at the connecting node has only translational degrees of freedom. This results in large relative motions between the rigid sheet and the deformable plate at the pin nodes. *Figure 3* shows the angular rotation about the z-axis at the connecting nodes for Case 1. The angular rotation at the pin node, node 8, is negative in response to the applied positive moment, which is the physically intuitive result.

Input files

Abaqus/Standard analysis

rigcon1_std.inp	Case 1.
rigcon2_std.inp	Case 2.
ver_rigcon2_std_thermexp.inp	Case 2 with temperature change.

Abaqus/Explicit analysis

rigcon1.inp	Case 1.
rigcon2.inp	Case 2.

Figures

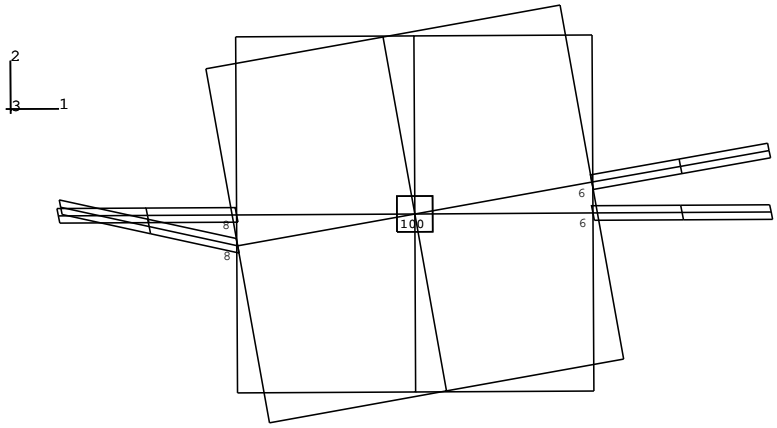


Figure 1: Original and final configurations for Case 1. Deformation magnification factor = 3.0.

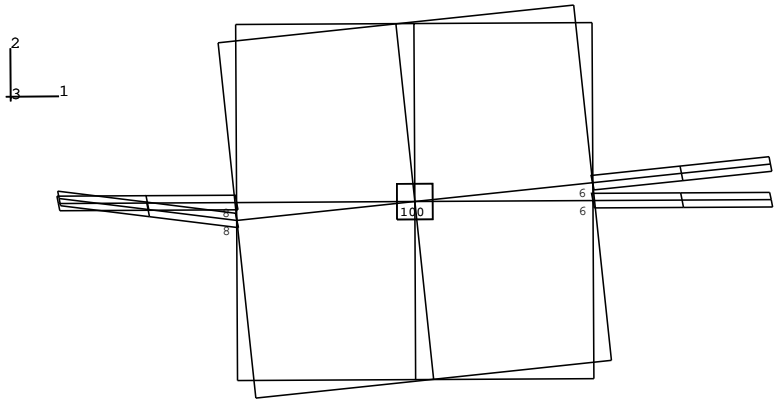


Figure 2: Original and final configurations for Case 2. Deformation magnification factor = 3.0.

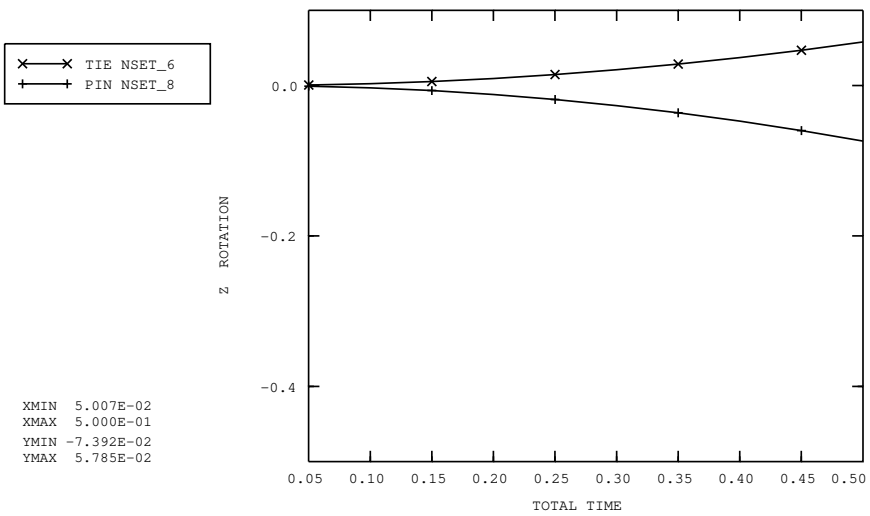


Figure 3: Rotation about the z-axis at the connecting nodes for Case 1.

Rigid body as an MPC

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

S4R

Features tested

Use of rigid body with TIE node set to define MPC between deformable elements.

Problem description

The model consists of two identical rectangular plates that lie parallel to the x–z plane and are initially separated by a distance of 1 m in the y-direction (see *Figure 1*). Each plate is modeled with S4R elements. Three pairs of nodes along the edges—node 1 of the bottom plate and node 10 of the top plate, node 9 of the bottom plate and node 90 of the top plate, node 4 of the bottom plate and node 40 of the top plate—are combined to form three distinct rigid bodies by including each pair in a TIE NSET. Concentrated loads of magnitude 1.0×10^5 N are applied in the positive z- and positive x-directions at nodes 20, 70, and 30 of the top plate. The results are compared to the solution of the corresponding MPC problem. In the MPC problem three BEAM-type MPCs are defined between the corresponding nodes of the top and bottom plates.

Results and discussion

The final configuration for the problem is shown in *Figure 2*. The bottom plate moves with the top plate so that the final configuration is similar to the original configuration except for a unified rotation and translation. This is because the rigid body TIE NSET constrains both the displacements and the rotations of the nodes that belong to it.

The results obtained using rigid body node sets closely match those obtained from solving the corresponding MPC problem. From *Figure 3*, *Figure 4*, and *Figure 5* it is clear that the leading characteristics of the solution—the u_x-displacement, the u_z-displacement, and the θ_x-rotation—are almost identical for the problem solved with rigid bodies and the corresponding BEAMMPC problem. The differences observed in the θ_x-rotation between Abaqus/Explicit and Abaqus/Standard are due to the different formulations used in the respective codes.

Input files

Abaqus/Standard analysis

rigmpc2_std.inp	Input file with rigid body definitions that include TIE NSETs.
rigmpc21_std.inp	Corresponding MPC problem.

Abaqus/Explicit analysis

rigmpc2.inp	Input file with rigid body definitions that include TIE NSETs.

rigmpc21.inp Corresponding MPC problem.

Figures

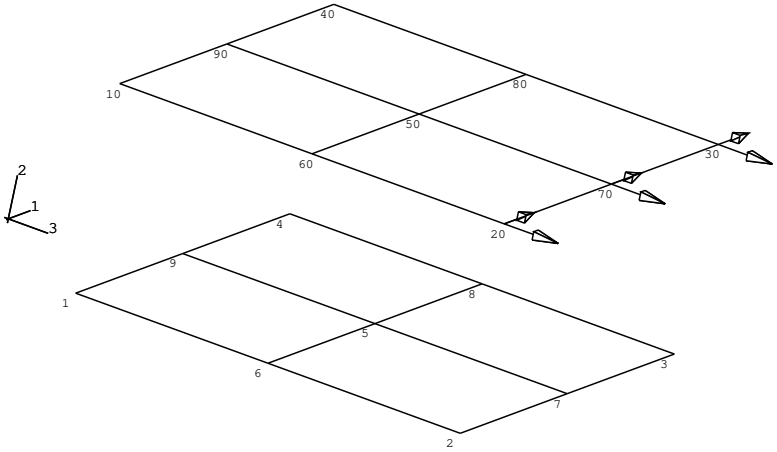


Figure 1: Original configuration.

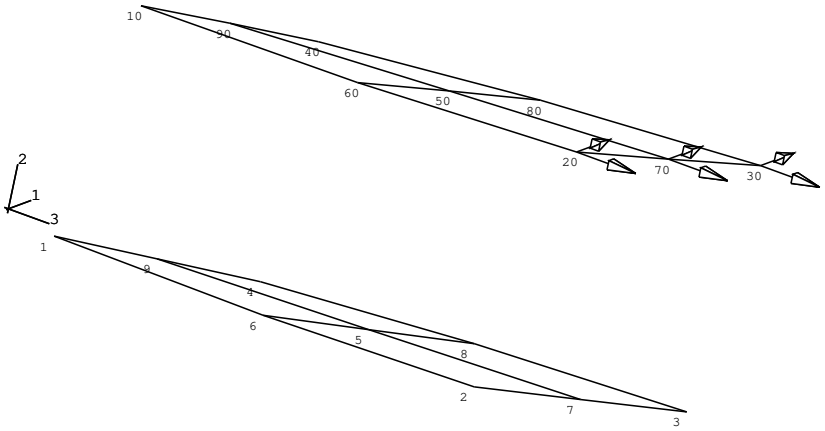


Figure 2: Final configuration.

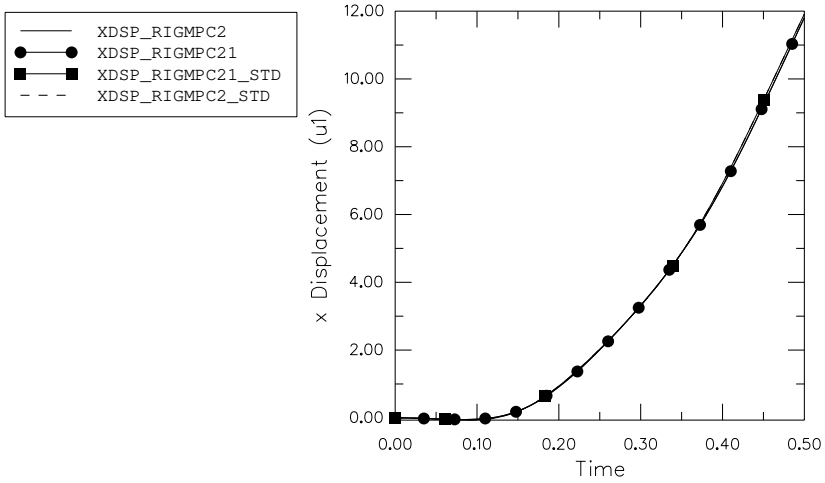


Figure 3: u_x-displacement versus time at node 9.

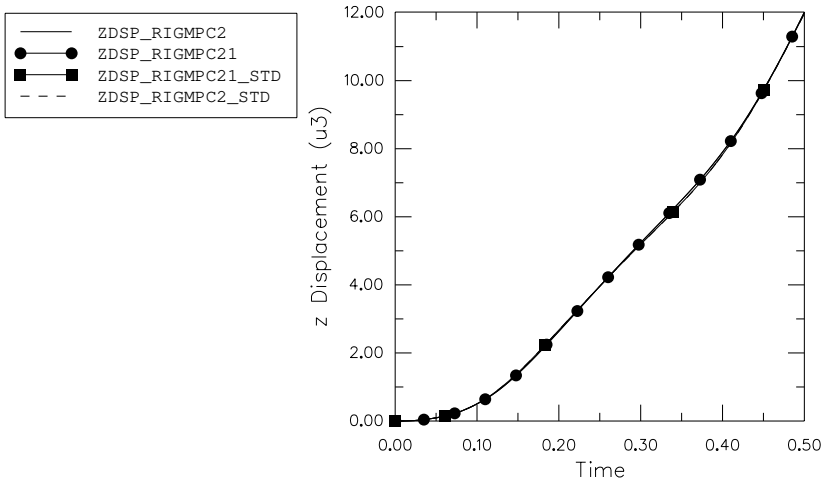


Figure 4: u_z-displacement versus time at node 9.

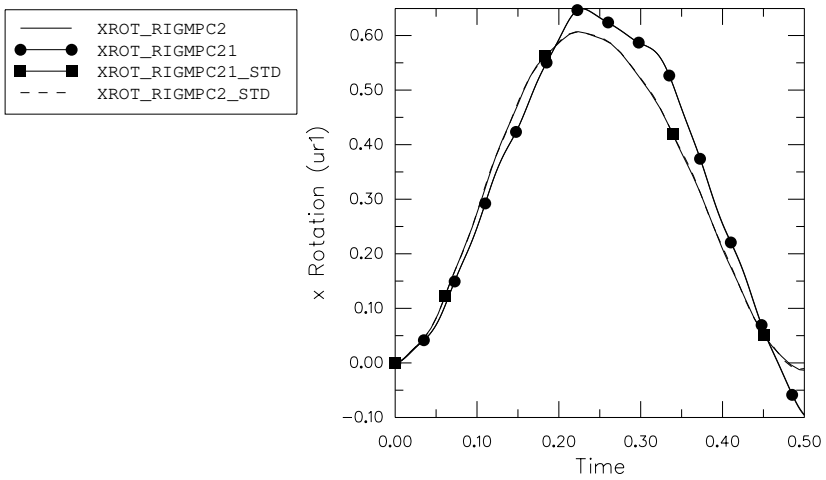


Figure 5: θ_x-rotation versus time at node 9.

Rigid body constraint

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

S4R

Features tested

Use of rigid bodies with TIE and PIN node sets to define boundary conditions for a deformable body.

Problem description

Rigid body node sets are defined to contain all nodes along the edges of a rectangular plate modeled with shell elements. The rigid body reference node is constrained against all rotations and u_x- and u_y-displacements. A saw-tooth velocity pattern acting in the z-direction is applied at the reference node of the rigid body. Starting at 0 m/s, the velocity is ramped down to −10 m/s at time 2.0×10^{-3} s and is ramped back to 0 m/s at time 6.0×10^{-3} s. Thereafter, the analysis is continued up to time 48.0×10^{-3} s. The following three cases are considered:

1. A rigid body TIE NSET is defined to contain all the edge nodes. The results are compared to the solution of the same problem with the rigid body TIE NSET replaced with equivalent boundary conditions applied at the edge nodes.
2. A rigid body PIN NSET is defined to contain all the edge nodes. The results are compared to the solution of the same problem with the rigid body PIN NSET replaced with equivalent boundary conditions applied at the edge nodes.
3. A rigid body TIE NSET is defined to contain all the nodes along two opposite edges of the plate. The remaining edge nodes are included in a PIN NSET. The results are compared to the solution of the same problem with the rigid body TIE and PIN node sets replaced with equivalent boundary conditions applied at the edge nodes.

Results and discussion

The plate displaces in response to the applied velocities at the boundary nodes and continues vibrating after the velocities at the boundary nodes have been ramped down to zero. The time variation of the u_z-displacement at node 205 at the center of the plate is plotted in *Figure 1* for Case 1. Following an initial lag, the center node vibrates in response to the boundary motion. The solution obtained using rigid body TIE NSET is found to match closely with the results of the same problem solved with the rigid body TIE NSET replaced by equivalent boundary conditions specified directly at the edge nodes. Similar conclusions can be drawn from *Figure 2* for Case 2 and *Figure 3* for Case 3.

Input files

Abaqus/Standard analysis

rigboun1_std.inp	Case 1.

rigboun1bc_std.inp	Comparison test of Case 1.
rigboun2_std.inp	Case 2.
rigboun2bc_std.inp	Comparison test of Case 2.
rigboun3_std.inp	Case 3.
rigboun3bc_std.inp	Comparison test of Case 3.

Abaqus/Explicit analysis

rigboun1.inp	Case 1.
rigboun1bc.inp	Comparison test of Case 1.
rigboun2.inp	Case 2.
rigboun2bc.inp	Comparison test of Case 2.
rigboun3.inp	Case 3.
rigboun3bc.inp	Comparison test of Case 3.

Figures

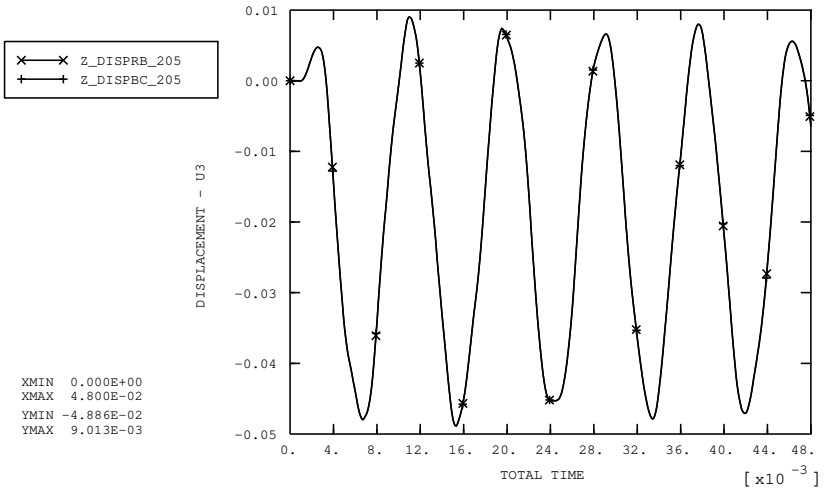


Figure 1: u_z-displacement at node 205 versus time for Case 1.

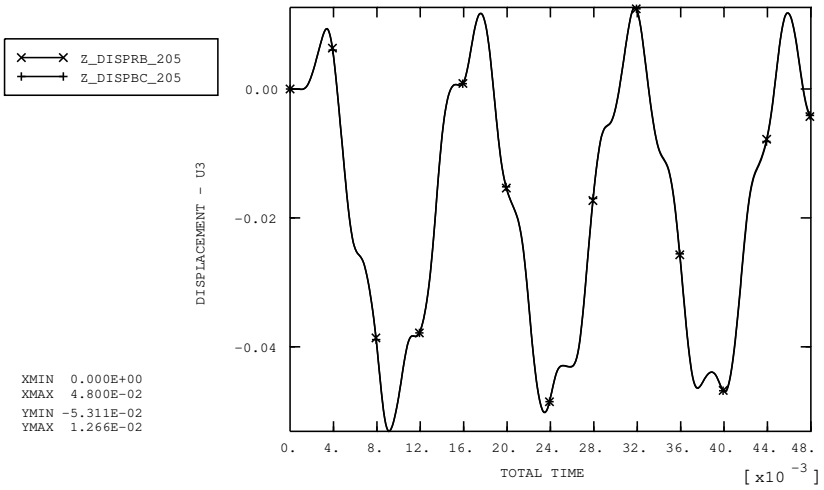


Figure 2: u_z-displacement at node 205 versus time for Case 2.

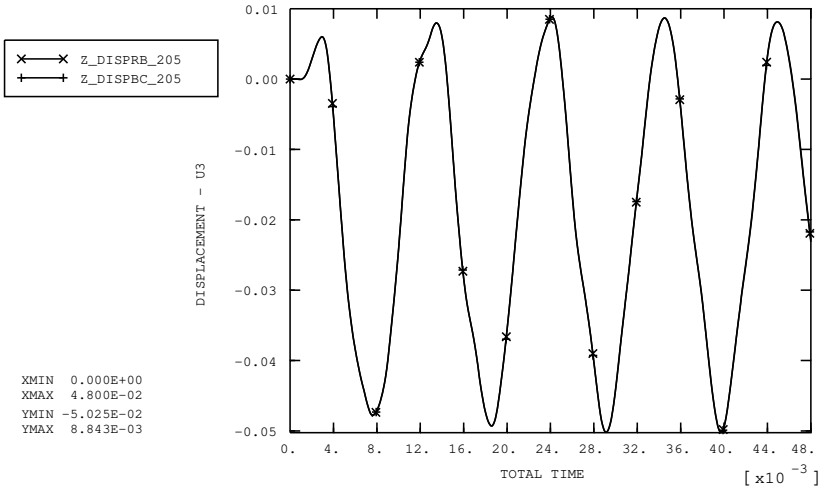


Figure 3: u_z-displacement at node 205 versus time for Case 3.

Including deformable element types in a rigid body

Products: Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D8R B31 S4R T3D2 F3D4

Features tested

Defining deformable elements as part of a rigid body.

Problem description

This example, which is similar to *Tennis racket and ball*, simulates the oblique impact of a tennis racket onto a stationary ball. The frame of the tennis racket is assumed to be rigid and is modeled using solid and structural elements of type C3D8R, B31, and S4R as part of a rigid body.

The strings on the tennis racket are modeled using T3D2 truss elements. The details of the material model used for the strings can be found in *Tennis racket and ball*. Initial tension is specified for the strings. The tennis ball is modeled as a sphere using S4R elements and is assumed to be made of rubber. The air in the tennis ball is modeled using the surface-based fluid cavity capability. A coefficient of friction is specified between the ball and the strings. In this example the ball is initially at rest, and the racket impacts the ball at 6.706 m/sec (264 in/sec) at an angle of 15°. The density of the elements representing the racket is chosen such that the mass of the racket is nearly 10 times that of the ball.

The complete model is shown in *Figure 1*.

Results and discussion

Figure 1 shows the position of the ball with respect to the strings in the undeformed configuration. The deformed shapes at different stages of the analysis are shown in *Figure 2* through *Figure 4*. The tennis racket frame can be seen to be moving as a rigid body, rotating slightly due to the distance between the point of impact and the racket center of mass. A deformation magnification factor of two has been used in plotting the figures.

Input files

tennis_rig.inp	Analysis using contact pair approach.
tennis_rig_gcont.inp	Analysis using the general contact capability.
tennis_rig1.inp	External file referenced by both analyses.
tennis_rig2.inp	External file referenced by both analyses.
tennis_rig3.inp	External file referenced by both analyses.
tennis_rig4.inp	External file referenced by both analyses.

Figures

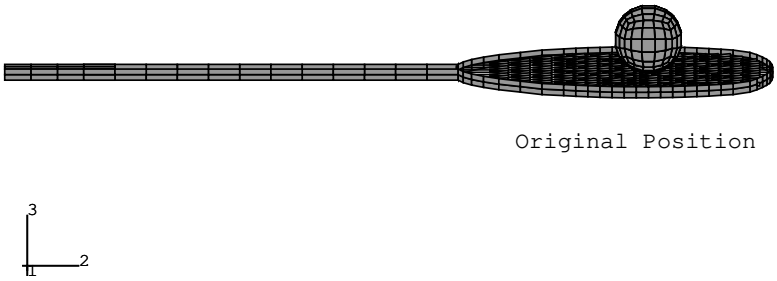


Figure 1: Original position of racket and ball.

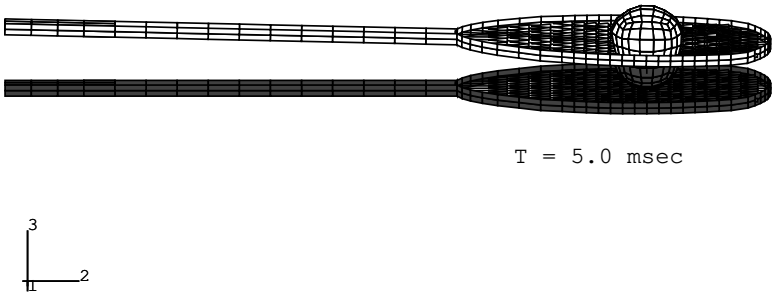


Figure 2: Configuration at 5 milliseconds.

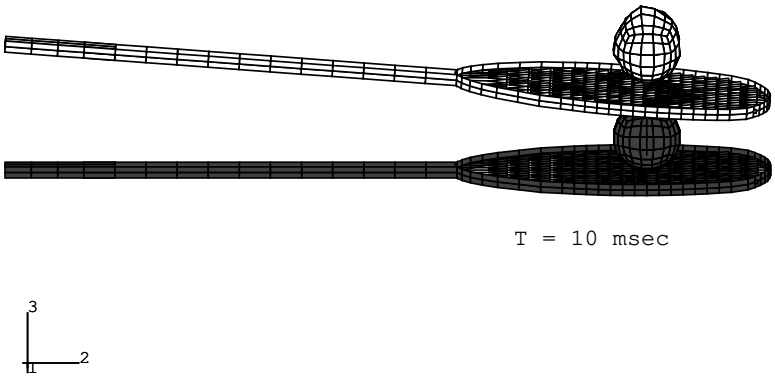


Figure 3: Configuration at 10 milliseconds.

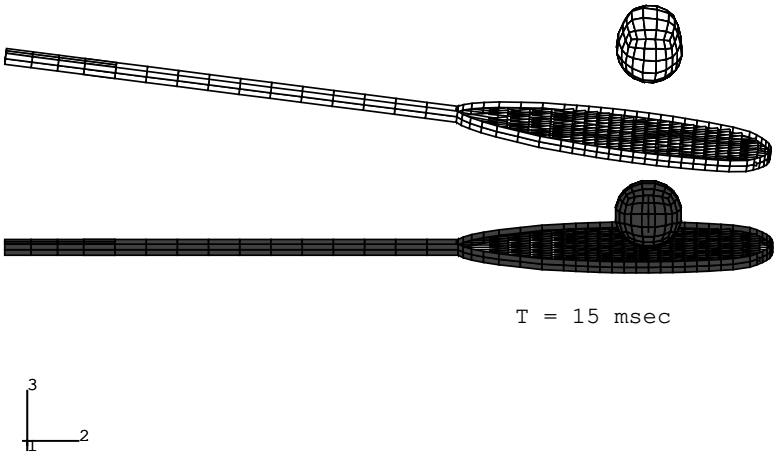


Figure 4: Configuration at 15 milliseconds.

Coupling constraints

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Features tested

This section provides basic verification tests for coupling, kinematic, and distributing constraints.

Kinematic coupling constraints

Features tested

Various types of kinematic coupling connections are tested.

Problem description

Problems *xcouplingk_std_beam.inp*, *xcouplingk_xpl_beam2d.inp*, *xcouplingk_std_bem3.inp*, and *xcouplingk_xpl_beam3d.inp* impose rigid beam constraints using coupling constraints.

Problems *xcouplingk_std_revolute.inp* and *xcouplingk_xpl_revolute.inp* test the finite rotation revolute behavior of the kinematic coupling constraint when only two rotational degrees of freedom are constrained.

Problems *xcouplingk_std_universal.inp* and *xcouplingk_xpl_universal.inp* test the finite rotation universal behavior of the kinematic coupling constraint when only one rotational degree of freedom is constrained.

Results and discussion

Comparisons with equivalent beam MPC and equivalent revolute and universal MPC problems show that using coupling constraints yields identical behavior.

Input files

Abaqus/Standard input files

xcouplingk_std_beam.inp	Equivalent to MPC type beam in a plane.
ver_xcouplingk_std_beam_thermexp.inp	Equivalent to MPC type beam in a plane with thermal expansion.
xcouplingk_std_bem3.inp	Equivalent to MPC type beam in space.
xcouplingk_std_revolute.inp	Test of the revolute rotational behavior of the kinematic coupling constraint.
xcouplingk_std_universal.inp	Test of the universal rotational behavior of the kinematic coupling constraint.

Abaqus/Explicit input files

xcouplingk_xpl_beam2d.inp	Equivalent to MPC type beam in a plane.
xcouplingk_xpl_beam3d.inp	Equivalent to MPC type beam in space.
xcouplingk_xpl_revolute.inp	Test of the revolute rotational behavior of the kinematic coupling constraint.
xcouplingk_xpl_universal.inp	Test of the universal rotational behavior of the kinematic coupling constraint.

Kinematic coupling constraints with local coordinate systems

Features tested

The kinematic coupling constraint with a local coordinate system applied at the coupling nodes is verified.

Problem description

Figure 1 shows the geometry for these problems .

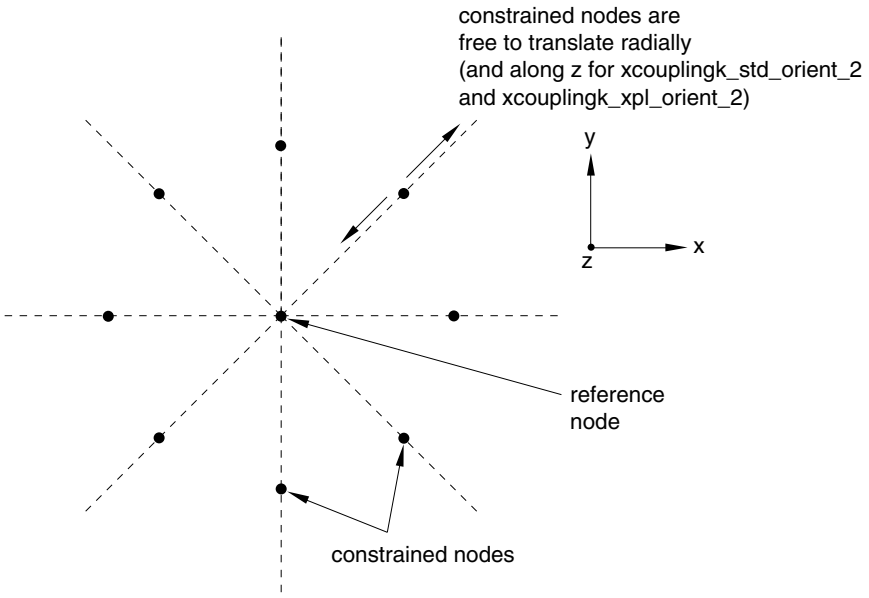


Figure 1: Geometry to test local orientation definitions.

In these tests the center node is the reference node, and the perimeter nodes are the coupling nodes. Four separate coupling definitions that share the same reference node are defined. Each coupling definition defines the local coordinate system using a different orientation system: cylindrical, rectangular, spherical, and, for the Abaqus/Standard analyses, a system defined by user subroutine *ORIENT*. In all cases, the resulting local constraint basis directions coincide with the local directions of a cylindrical coordinate system whose axis is normal to the plane containing the nodes and passes through the reference node.

In problems *xcouplingk_std_orient_1.inp* and *xcouplingk_xpl_orient_1.inp* the kinematic coupling constrains all but the radial degree of freedom at the coupling nodes. Linear springs to ground (SPRING1) for the Abaqus/Standard analyses and connector elements to ground (CONN3D2) with linear elastic connector behavior for the Abaqus/Explicit analyses are attached to all coupling nodes and act in the x- and y-directions. The reference node is then rotated 2π radians about the z-axis.

In problems *xcouplingk_std_orient_2.inp* and *xcouplingk_xpl_orient_2.inp*the kinematic coupling constrains the circumferential degree of freedom only. Linear springs to ground (SPRING1) for the Abaqus/Standard analyses and connector elements to ground (CONN3D2) with linear elastic connector behavior for the Abaqus/Explicit analyses are attached to all coupling nodes and act in the *x*-, *y*-, and *z*-directions. The reference node is then rotated 2π about *x*-axis.

Results and discussion

These tests result in motion of the constrained nodes, under action of the linear springs, as the reference node rotates. For tests *xcouplingk_std_orient_1.inp* and *xcouplingk_xpl_orient_1.inp*this motion remains on the local radius passing through the node at all increments. For tests *xcouplingk_std_orient_2.inp* and *xcouplingk_xpl_orient_2.inp* this motion remains in the plane defined by the original configuration local radius and the global *z*-direction as this plane rotates according to the motion prescribed at the reference node.

Input files

Abaqus/Standard input files

xcouplingk_std_orient_1.inp	Test of local orientation and the release of a single translational degree of freedom.
xcouplingk_std_orient_1.f	User subroutine *ORIENT* for xcouplingk_std_orient_1.inp.
xcouplingk_std_orient_2.inp	Test of local orientation and the release of a two translational degrees of freedom.
xcouplingk_std_orient_2.f	User subroutine *ORIENT* for xcouplingk_std_orient_2.inp.

Abaqus/Explicit input files

xcouplingk_xpl_orient_1.inp	Test of local orientation and the release of a single translational degree of freedom.
xcouplingk_xpl_orient_2.inp	Test of local orientation and the release of two translational degrees of freedom.

Internal sorting of kinematic coupling constraints

Features tested

The internal sorting of kinematic coupling constraints when used in conjunction with MPC definitions is verified.

Problem description

The model consists of an axial arrangement of 20 shell elements. These elements are tied together using a combination of kinematic coupling constraints as well as MPCs. The constraints are defined such that the kinematic coupling reference node appears after the constraint definitions that are eliminated degrees of freedom on that node; thus, constraint sorting is required. The structure is clamped on one end, and a concentrated axial load is applied on the other end.

Results and discussion

The test results in an internal sorting of kinematic coupling definitions and MPCs so that the proper elimination order is achieved.

Input files

Abaqus/Standard input file

xcouplingk_std_sort.inp	Test internal sorting of kinematic coupling constraints.

Abaqus/Explicit input file

xcouplingk_xpl_sort.inp	Test internal sorting of kinematic coupling constraints.

Distributing coupling constraints with user-specified weights

Features tested

The distributing coupling constraint is tested by using coupling and distributing constraints with user-specified distributing weight factors. Geometric linear and nonlinear tests are performed.

Problem description

Model: The initial starting geometry for each test is shown in *Figure 2*. For the geometric linear test, for Abaqus/Standard, each coupling node is connected by a spring to ground (SPRING1) in each direction. In the geometrically nonlinear test in Abaqus/Standard, each coupling node is connected by a dashpot to ground (DASHPOT1) in each direction, and an axial spring element (SPRINGA) connects each pair of coupling nodes. In the geometrically nonlinear test in Abaqus/Explicit, each coupling node is connected by a connector to ground (CONN3D2) with damping behavior specified in each direction, and a connector element with specified elastic behavior connects each pair of coupling nodes. The reference node for the coupling constraint is node 10.

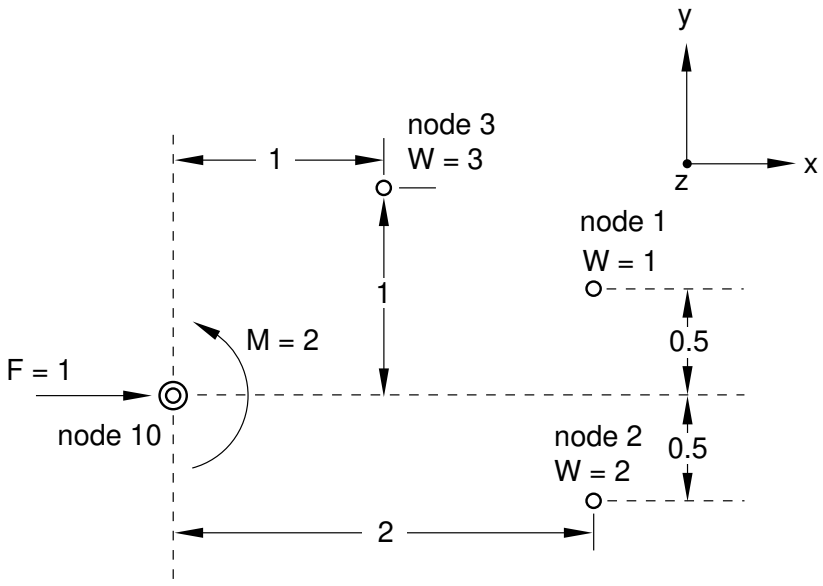


Figure 2: Initial starting geometry.

Linear behavior

Properties:

The spring stiffnesses are 100, 200, and 300 for degrees of freedom 1, 2, and 3, respectively, for the springs connected to all coupling nodes. The distributing weight factors are 1, 2, and 3 for nodes 1, 2, and 3, respectively.

Loading:

Step 1	The force at the reference node is 1.0 in the *x*-direction. The moment at the reference node is 2.0 about the *z*-axis.
Step 2	The force at the reference node is 1.0 in the *y*-direction. The moment at the reference node is 2.0 about the *x*-axis.
Step 3	The force at the reference node is 1.0 in the *z*-direction. The moment at the reference node is 2.0 about the *y*-axis.
Step 4	Frequency extraction.
Step 5	Transient modal dynamic step with a load, $F_x = 1.0\sin 2\pi t$, applied to the reference node.
Step 6	Mode-based steady-state dynamic step with a load, $F_x = 1.0$, applied to the reference node.

Nonlinear behavior

Properties:

The dashpot damping coefficients are 100, 200, and 300 for degrees of freedom 1, 2, and 3, respectively, for the dashpots connected to all coupling nodes. The axial springs connecting the coupling nodes each have a spring constant of 1.0×10^8. The distributing weight factors are 1, 2, and 3 for nodes 1, 2, and 3, respectively.

Prescribed reference node motion for Abaqus/Standard:

Step 1	Total rotation of 2π about the *z*-axis. Translation $x = \sin 2\pi t$.
Step 2	Total rotation of 2π about the *y*-axis. Translation $z = \sin 2\pi t$.
Step 3	Total rotation of 2π about the *x*-axis. Translation $y = \sin 2\pi t$.
Step 4	Direct-integration dynamic step with a total rotation of 2π about the *z*-axis. Translation $x = \sin 2\pi t$.

Prescribed reference node motion for Abaqus/Explicit:

Step 1	Total rotation of 2π about the *z*-axis. Translation $x = \sin 2\pi t$.
Step 2	Total rotation of 2π about the *y*-axis. Translation $z = \sin 2\pi t$.
Step 3	Total rotation of 2π about the *x*-axis. Translation $y = \sin 2\pi t$.
Step 4	Total rotation of 2π about the *z*-axis. Translation $x = \sin 2\pi t$.

Results and discussion

In all tests the load distribution among coupling nodes adheres to the relation

$$\mathbf{F}^n = \widehat{w}^n \left(\mathbf{F}^R + \mathbf{T}^{-1} \left(\mathbf{M}^R + \mathbf{r}^R \times \mathbf{F}^R \right) \times \mathbf{r}^n \right),$$

where $\mathbf{F}^n$ is the force distribution at the coupling nodes, $\mathbf{F}^R$ and $\mathbf{M}^R$ are the force and moment at the reference node, $\widehat{w}^n$ are the normalized distributing weight factors, $\mathbf{T}$ is the coupling node arrangement inertia tensor, and $\mathbf{r}^R$ and $\mathbf{r}^n$ are the positions of the reference and coupling nodes relative to the coupling node arrangement centroid, respectively. See *Distributing coupling constraints* for a more detailed description of this load distribution.

Input files

Abaqus/Standard input files

xcouplingd_std_geomlinear.inp	Distributing coupling for geometric linear case.
xcouplingd_std_geomnonlinear.inp	Distributing coupling for geometric nonlinear case.

Abaqus/Explicit input file

xcouplingd_xpl_geomnonlinear.inp	Distributing coupling for geometric nonlinear case.

Default distributing weight factors

Elements tested

B21 B22
C3D8 C3D8R C3D10M C3D20 C3D27
CAX4 CAX4R CAX8
CPE4 CPE4R CPE8
S3R S4 S8R S9R5
CSS8

Features tested

The default distributing weight factors for a distributing coupling constraint are verified. The weight factors are based on the nodal tributary surface area at each coupling node.

Problem description

Various models consisting of either continuum, beam, or shell elements are used in this test. In all models a uniform surface load is applied via a reference node and a distributing coupling constraint. A nonuniform mesh density is used to verify that the proper tributary area is calculated. The reference node is located at the center of the loaded surface, offset in the normal direction.

Results and discussion

The displacements are equal to the displacements obtained if the model were loaded with a uniform pressure load, hence verifying that the proper distributing weights are calculated at the coupling nodes.

Input files

Abaqus/Standard input files

xcouplingd_std_tarea_b21.inp	Surface with underlying B21 elements.
xcouplingd_std_tarea_b22.inp	Surface with underlying B22 elements.
xcouplingd_std_tarea_c3d10m.inp	Surface with underlying C3D10M elements.
xcouplingd_std_tarea_c3d20.inp	Surface with underlying C3D20 elements.
xcouplingd_std_tarea_c3d27.inp	Surface with underlying C3D27 elements.
xcouplingd_std_tarea_c3d8.inp	Surface with underlying C3D8 elements.
xcouplingd_std_tarea_cax4.inp	Surface with underlying CAX4 elements.
xcouplingd_std_tarea_cax8.inp	Surface with underlying CAX8 elements.
xcouplingd_std_tarea_cpe4.inp	Surface with underlying CPE4 elements.
xcouplingd_std_tarea_cpe8.inp	Surface with underlying CPE8 elements.
xcouplingd_std_tarea_s3r.inp	Surface with underlying S3R elements.
xcouplingd_std_tarea_s4.inp	Surface with underlying S4 elements.
xcouplingd_std_tarea_s8r.inp	Surface with underlying S8R elements.
xcouplingd_std_tarea_s9r5.inp	Surface with underlying S9R5 elements.
xcouplingd_std_tarea_css8.inp	Surface with underlying CSS8 elements.

Abaqus/Explicit input files

xcouplingd_xpl_tarea_b21.inp	Surface with underlying B21 elements.
xcouplingd_xpl_tarea_c3d10m.inp	Surface with underlying C3D10M elements.
xcouplingd_xpl_tarea_c3d8r.inp	Surface with underlying C3D8R elements.
xcouplingd_xpl_tarea_cax4r.inp	Surface with underlying CAX4R elements.
xcouplingd_xpl_tarea_cpe4r.inp	Surface with underlying CPE4R elements.
xcouplingd_xpl_tarea_s4r.inp	Surface with underlying S4R elements.

Weighting method and influence region

Features tested

The calculation of distributing weights as outlined in *Coupling Constraints* when the optional weighting method and influence region are specified is verified. The use of coupling constraints at the part-instance level is also illustrated.

Problem description

A part is defined consisting of two rows of 20 CPE4R elements. Each element is a unit square. The coupling nodes are defined along the top surface. A reference node is created at the center of the top surface. The part is then instanced three times in the assembly definition. For each part instance a coupling constraint with a different influence region is defined. The first part instance has an infinite influence radius; that is, all nodes defined on the surface will be included in the coupling definition. The second part instance uses an influence radius of 5.5, and the third part instance uses an influence radius of 0.5. A concentrated load is applied to each reference node. Input files are provided for each weighting scheme: uniform, linear, quadratic, and cubic.

Results and discussion

The distributing weight factor calculations are verified to be according to the description provided in *Coupling Constraints*. For the first instance, all nodes belonging to the facets are included in the coupling definition. For the second instance, the nodes of six facets adjacent to the reference node are included in the coupling definition. In this case the facet farthest away from the reference node (on either side) uses a facet participation factor of 0.5, since only part of element surface facet is included in the influence region. For the third case, the nodes of the adjacent facets to the reference node are included in the coupling definition. In this case each facet has a participation factor of 0.5, since only part of the element surface facet is included in the influence region.

Input files

Abaqus/Standard input files

xcouplingd_std_wgt_uniform.inp	Distributing coupling with a uniform weighting method.
xcouplingd_std_wgt_linear.inp	Distributing coupling with a linear weighting method.
xcouplingd_std_wgt_quadratic.inp	Distributing coupling with a quadratic weighting method.
xcouplingd_std_wgt_cubic.inp	Distributing coupling with a cubic weighting method.

Abaqus/Explicit input files

xcouplingd_xpl_wgt_uniform.inp	Distributing coupling with a uniform weighting method.
xcouplingd_xpl_wgt_linear.inp	Distributing coupling with a linear weighting method.
xcouplingd_xpl_wgt_quadratic.inp	Distributing coupling with a quadratic weighting method.
xcouplingd_xpl_wgt_cubic.inp	Distributing coupling with a cubic weighting method.

Collinear coupling node arrangement

Features tested

A pathological situation in which all coupling nodes are collinear for a distributing coupling constraint and the moment applied at the reference node is not transmitted by the constraint is tested.

Problem description

The geometry is shown in *Figure 3*.

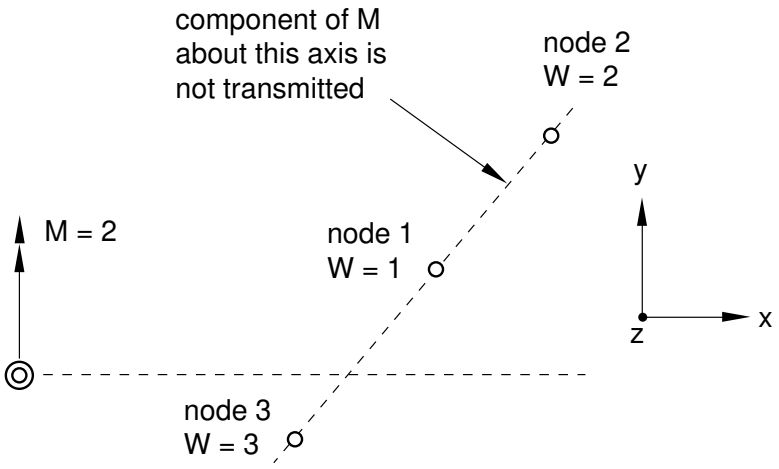


Figure 3: Collinear coupling node arrangement.

Results and discussion

The distributing coupling constraint connects a single reference node that has translational and rotational degrees of freedom to a collection of coupling nodes that have only translational degrees of freedom. Thus, when the coupling nodes are collinear in a three-dimensional analysis, a situation can arise where the moments applied to the reference node are not transmitted. In such a case Abaqus will print a warning message specifying the axis about which the moments are not transmitted.

Input files

Abaqus/Standard input file

xcouplingd_std_colinear_nodes.inp	Distributing coupling with collinear coupling nodes.

Abaqus/Explicit input file

xcouplingd_xpl_colinear_nodes.inp	Distributing coupling with collinear coupling nodes.

Moment release for distributing coupling

Features tested

A series of linear and nonlinear analyses are performed demonstrating the ability of the distributing coupling constraints to release the rotation constraints between the reference node and the coupling nodes about user-specified axes.

Problem description

This example consists of both a two-dimensional and three-dimensional test.

In the two-dimensional test, two separate models are defined. Each model consists of a single CPE4 element with one face coupled to a reference node with a distributing constraint. The opposite face of the CPE4 element is fixed. Beam elements are attached to the reference nodes for visualization purposes only. The first model uses the default coupling in which the rotation degree of freedom of the reference node is coupled to the solid surface

(the displacement degrees of freedom of the reference are always coupled to the surface with distributing constraints). The second model releases the rotation constraint. A series of boundary conditions are applied to the reference nodes simulating shear, tension, and bending (in various linear and nonlinear steps).

In the three-dimensional test, eight separate models are defined. Each model consists of a single C3D8 element with one face coupled to a reference node with a distributing constraint. The opposite faces of the C3D8 elements are fixed. Beam elements are attached to the reference nodes for visualization purposes only. The first model uses the default coupling in which all three rotation degrees of freedom of the reference node are coupled to the solid surface. The next three models respectively release the rotation constraint in the 1, 2, and 3 directions. The final four models are identical to the first four, except that the rotation constraint directions are specified. A series of boundary conditions are applied to the reference nodes simulating shear, tension, and bending (in linear and nonlinear steps).

Results and discussion

The results clearly show that both coupling definitions in both two and three dimensions are being applied properly.

Input files

Abaqus/Standard input files

xcouplingd_std_release_2d.inp	Two-dimensional examples of distributing coupling with the moment constraints released.
xcouplingd_std_release_3d.inp	Three-dimensional examples of distributing coupling with the moment constraints released.

Dimensional coupling

Features tested

A series of linear analyses are performed demonstrating the ability of the distributing coupling constraints to provide accurate dimensional coupling of beam elements to shell and solid elements.

Problem description

This example consists of two sets of tests in which a pipe is modeled with beam and shell elements and with beam and continuum elements.

The pipe analyzed with beam and shell elements has a length of 0.8 m, an outside radius of 0.1 m, and a thickness of 0.01 m. The material has a Young's modulus of 200 GPa and a Poisson's ratio of 0.3. Half of the pipe is modeled with beam elements and the other half is modeled with shell elements (see *Figure 4*(a)). The beam node closest to the shell model is defined as the reference node for the distributing coupling constraint. An element-based edge surface is defined on the shell model, which is coupled to the reference node. The coupled model is subjected to four linear loading conditions simulating: (1) twist about the pipe axis, (2) axial stretch along the pipe axis, (3) pure bending about the x-axis, and (4) shear loading. The four load conditions are applied in a single linear step as four load cases. Two models are analyzed: one with linear beam and shell elements and one with quadratic beam and shell elements.

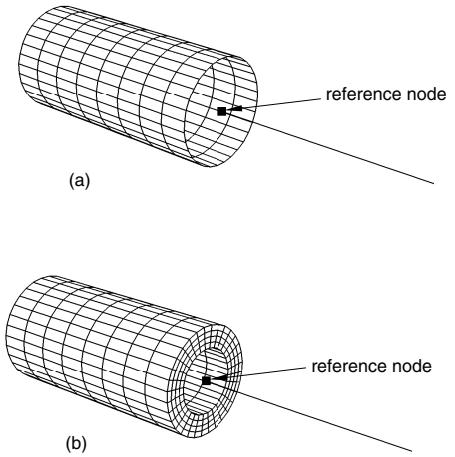


Figure 4: Dimensional coupling examples: (a) beam-to-shell coupling model; (b) beam-to-solid coupling model.

The pipe analyzed with beam and continuum elements has a length of 0.8 m, an outside radius of 0.1 m, and a thickness of 0.04 m. The material has a Young's modulus of 200 GPa and a Poisson's ratio of 0.3. Half of the pipe is modeled with beam elements and the other half is modeled with continuum elements (see *Figure 4*(b)). The beam node closest to the continuum model is defined as the reference node for the distributing coupling constraint. An element-based surface is defined on the continuum model, which is coupled to the reference node. The coupled model is subjected to four linear loading conditions simulating: (1) twist about the pipe axis, (2) axial stretch along the pipe axis, (3) pure bending about the *x*-axis, and (4) shear loading. The four load conditions are applied in a single linear step. Two models are analyzed: one with linear beam and continuum elements and one with quadratic beam and continuum elements.

Results and discussion

The resulting stress fields in the shell and solid models show minimal distortion at the coupling interface, indication that the dimensional coupling is modeled accurately.

Input files

Abaqus/Standard input files

xcoupling_beamtoshell_lin.inp	Coupling a beam model to a shell model using linear beam and shell elements.
xcoupling_beamtoshell_quad.inp	Coupling a beam model to a shell model using quadratic beam and shell elements.
xcoupling_beamtosolid_lin.inp	Coupling a beam model to a continuum model using linear beam and continuum elements.
xcoupling_beamtosolid_quad.inp	Coupling a beam model to a continuum model using quadratic beam and continuum elements.

Structural coupling

Features tested

A series of analyses are performed demonstrating the structural coupling capability of small distributing coupling constraints.

Problem description

Four different models, each with two small distributing couplings, are analyzed. In the first model two small square plates are coupled together with a BEAM connector. The connector nodes are coupled to the two small surfaces using structural distributing couplings. One plate is kept fixed, while the other is pulled upward (pried open) on one side. In the second model, the same plates are pulled upward from all sides. In the third model two circular plates are fastened together by placing a BEAMMPC between the reference nodes of two structural distributing couplings spanning two small patches on the two plates. The plates are then subjected to relative shear motion. In the fourth model two U-shaped shell specimens are connected in a fashion similar to that in the second model. The lower specimen is fixed, while the upper specimen is lifted and pried open simultaneously.

For comparison in Abaqus/Explicit, similar models are created to use continuum distributing coupling and fasteners.

Results and discussion

The resulting deformed shapes match the expectations. More important, if structural coupling is used, contact between the plates does not occur in the area close to the fastener, as expected. By contrast, contact does occur if continuum distributing couplings are used.

Input files

Abaqus/Explicit input files

couplingstruct_pry_dist_xpl.inp	First model described above with structural coupling.
couplingcont_pry_dist_xpl.inp	First model described above with continuum coupling.
couplingstruct_pry_fast_xpl.inp	First model described above with structural coupling via fasteners.
couplingstruct_pull_dist_xpl.inp	Second model described above with structural coupling.
couplingcont_pull_dist_xpl.inp	Second model described above with continuum coupling.
couplingstruct_pull_fast_xpl.inp	Second model described above with structural coupling via fasteners.
couplingstruct_circle_dist_xpl.inp	Third model described above with structural coupling.
couplingcont_circle_dist_xpl.inp	Third model described above with continuum coupling.
couplingstruct_circle_fast_xpl.inp	Third model described above with structural coupling via fasteners.
couplingstruct_utestrig_dist_xpl.inp	Fourth model described above with structural coupling.
couplingcont_utestrig_dist_xpl.inp	Fourth model described above with continuum coupling.
couplingstruct_utestrig_fast_xpl.inp	Fourth model described above with structural coupling via fasteners.

Abaqus/Standard input files

couplingstruct_pry_dist_std.inp	First model described above with structural coupling.
couplingstruct_pry_fast_std.inp	First model described above with structural coupling via fasteners.
couplingstruct_pull_dist_std.inp	Second model described above with structural coupling.
couplingstruct_pull_fast_std.inp	Second model described above with structural coupling via fasteners.
couplingstruct_circle_dist_std.inp	Third model described above with structural coupling.
couplingstruct_circle_fast_std.inp	Third model described above with structural coupling via fasteners.
couplingstruct_utestrig_dist_std.inp	Fourth model described above with structural coupling.
couplingstruct_utestrig_fast_std.inp	Fourth model described above with structural coupling via fasteners.

Auto convert single facet distributing coupling to tie

Features tested

Abaqus/Explicit converts a distributing coupling to a surface-based tie constraint if the cloud nodes of the distributing coupling correspond to a single facet of an element unless the reference node of the coupling is connected to a connector element

Problem description

The model specifications involve distributing couplings with reference nodes corresponding to nodes of solid elements and cloud nodes corresponding to nodes of shell elements. Many of the distributed couplings specified in the model have cloud nodes limited to one element face; Abaqus/Explicit automatically converts these distributed couplings to node-to-face surface-based tie constraints involving the same set of nodes. The motivation for this conversion of the model is that Abaqus/Explicit automatically adds mass to nodes that act as cloud nodes in multiple distributing couplings, but Abaqus/Explicit does not automatically add mass to nodes acting as main nodes in multiple surface-based tie constraints.

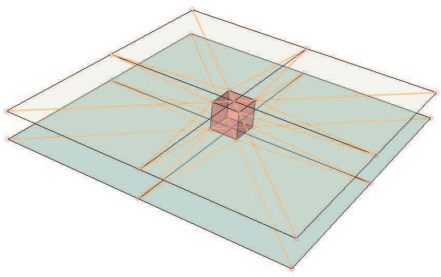


Figure 5: A fastener construct involving a hexahedral element representing the faster stiffness.

Results and discussion

Distributed couplings with cloud nodes associated with a single element face have been internally converted to node-to-face surface-based tie constraints, and the amount of mass artificially added to the model is reduced because fewer nodes participate in multiple distributed couplings.

Input files

DcoupOverlapped.inp	Two plates fastened using a hexahedral element acting as a fastener and distributed couplings attached to plates. The distributed couplings overlap.
DcoupTieConvert.inp	Various distributed couplings in fastener application .
xcouplingd_xpl_tieconvert.inp	Single facet distributing coupling attached to shells.

Rotational coupling for distributing coupling constraint for collinear cloud nodes

Features tested

If cloud nodes of a distributing coupling constraint are collinear, the component of rotation of the reference node about the direction of collinearity will participate in the distributed coupling constraint only if cloud nodes have rotational degrees of freedom that participate in the distributing coupling.

Problem description

The following tests verify that appropriate constraints are formed for distributing couplings with collinear cloud nodes with rotational degrees of freedom that are allowed to participate in the coupling (see *Controlling Whether Cloud Node Rotations Influence Reference Node Rotation*).

Distributing coupling problems `HingeDcoupRotCoupling.inp` (Abaqus/Explicit) and `std_HingeDcoupRotCoupling.inp` (Abaqus/Standard) have collinear cloud nodes attached to shell elements in a hinge-like arrangement. The reference node, if offset, forms the weighted center of the cloud nodes. The weighted center location of the distributing coupling has the average translation and rotation of the cloud nodes. In this case, the cloud nodes all have the same translation and rotation. The distributing coupling includes a rigid arm between the weighted center location and the reference node. The reference node is at a greater radius from the location of the hinge in this example, causing a larger resultant displacement at the reference node compared to the cloud nodes.

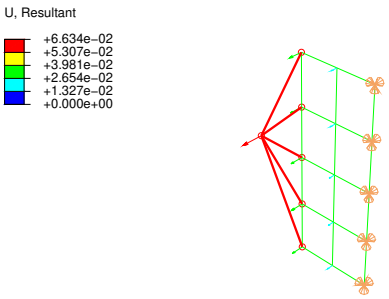


Figure 6: Results for the `HingeDcoupRotCoupling` model.

In `PlateDcoupRotCoupling.inp` (Abaqus/Explicit) and `std_PlateDcoupRotCoupling.inp` (Abaqus/Standard), a distributing coupling is used to connect four collinear cloud nodes of a plate to a reference node at the end of a beam. A rotational boundary condition is used to spin the beam about its axis, and the distributing coupling imposes that the collinear cloud nodes of the shell have the same average rotation as the reference node.

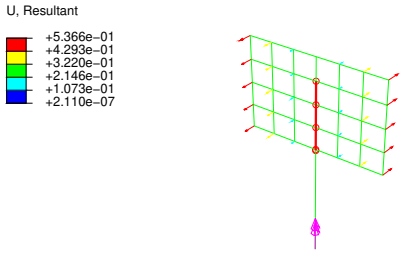


Figure 7: Results for the `PlateDcoupRotCoupling` model.

`automaticrotcoupling.inp` (Abaqus/Explicit) involves two collinear beams. A distributing coupling is used to connect collinear nodes of one beam with a reference node of the other beam. A torque load about the beam axis is applied to the lower beam. Distributing coupling leads to the matching axial rotations of both beams.

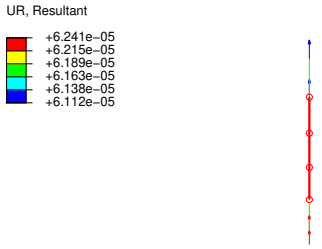


Figure 8: Results for the `automaticrotcoupling` model.

`rotcouplingcomparison.inp` (Abaqus/Explicit) involves two instances of a distributing coupling between a reference node at a point mass and four cloud nodes at nodes of a single shell element. Each of the cloud node systems is not collinear, so valid constraints are formed even if reference node rotations are coupled only to cloud node translations. Rotation of the reference node is coupled to cloud node translations and rotations for the case shown on the left, and rotation of the reference node is coupled only to cloud node translations for the other case. Similar results are obtained for both cases.

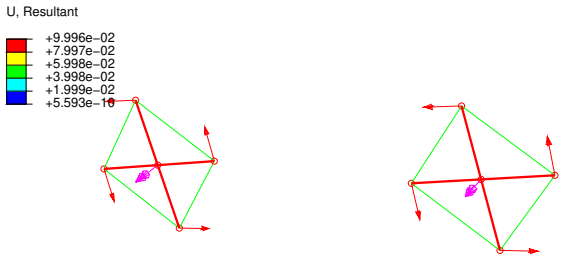


Figure 9: Results for the `rotcouplingcomparison` model.

Results and discussion

The results show that rotation parallel to the axis of collinearity of the cloud nodes is properly constrained by the distributing coupling for `HingeDcoupRotCoupling.inp` and `PlateDcoupRotCoupling.inp`. For `automaticrotcoupling.inp` rotational structural is chosen as the default. For `rotcouplingcomparison.inp` the solutions with rotational structural coupling and rotational continuum coupling show similar solutions when the cloud nodes are not collinear.

Input files

HingeDcoupRotCoupling.inp	Abaqus/Explicit analysis of a plate with distributing coupling that forms one side of a hinge-like construct. Rotational velocity is applied at the reference node.
std_HingeDcoupRotCoupling.inp	Abaqus/Standard analysis of a plate with distributing coupling that forms one side of a hinge-like construct. Rotational velocity is applied at the reference node.
PlateDcoupRotCoupling.inp	Abaqus/Explicit analysis of a plate and beam attached to each other with a distributing coupling that has collinear nodes. The plate is rotated, and the coupling twists the beam.
std_PlateDcoupRotCoupling.inp	Abaqus/Standard analysis of a plate and beam attached to each other with a distributing coupling that has collinear nodes. The plate is rotated, and the coupling twists the beam.
automaticrotcoupling.inp	Abaqus/Explicit analysis with distributing coupling with collinear nodes. The default option demonstrates that structural rotational coupling is used for collinear cloud nodes.
rotcouplingcomparison.inp	Comparison of two distributing couplings that do not have collinear nodes. Rotational structural coupling and rotational continuum coupling are used. Results show that using rotational structural coupling for when the cloud nodes are not collinear produces results that are very close to the one using continuum rotational coupling.

Kinematic coupling constraint

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

B22 B32 B33H MASS S4R SPRING1

Features tested

Various types of kinematic coupling connections are tested by selectively constraining degrees of freedom. Where tests are equivalent to existing multi-point constraint tests, references to those verification tests are made. Refer to *Multi-point constraints* for details of these tests.

Problem description

Problems *xkcbeam.inp* and *xkcbem3.inp* impose beam constraints using kinematic coupling constraints and are the same as the equivalent MPC problem. Problem *xkcrevo.inp* tests the finite-rotation revolute behavior of the kinematic coupling when only two rotational degrees of freedom are constrained. Problem *xkcuniv.inp* tests the finite-rotation universal behavior of the kinematic coupling when only one rotational degree of freedom is constrained.

The geometry for problems *xkccirc.inp* and *xkccirc2.inp* is shown in *Figure 1*.

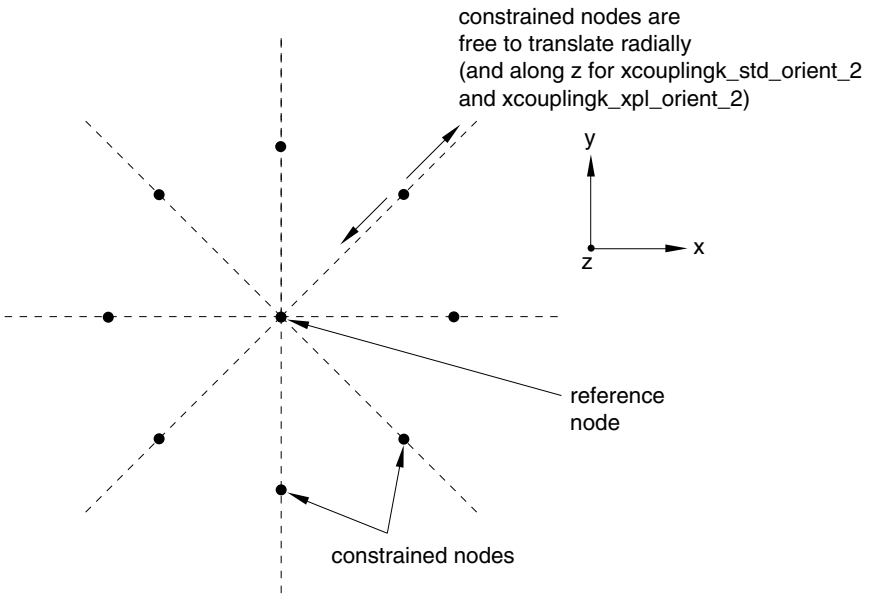


Figure 1: Geometry for xkccirc.inp and xkccirc2.inp.

In this test the center node is the kinematic coupling reference node, and the perimeter nodes are the coupling nodes. To verify the options for specifying local coordinate systems at these coupling nodes, the constraint shown

is created using four separate kinematic coupling definitions that share the center reference node. Each of these coupling definitions defines the local coordinate system using a different orientation system: cylindrical, rectangular, spherical, and a system defined in user subroutine *ORIENT*. In all cases the resulting local constraint basis directions coincide with the local directions of a cylindrical coordinate system whose axis is normal to the plane containing the nodes and passes through the reference node. Problem *xkccirc.inp* also includes nodal transformations at some nodes; this will have no effect on the constraints.

In the case of *xkccirc.inp* the kinematic coupling constrains all but the radial degree of freedom at the constrained nodes. Linear springs to ground (SPRING1) are attached to all constrained nodes and act in the x-direction. The reference node is then rotated $\mathbf{2\pi}$ about z during a static step.

In the case of *xkccirc2.inp* the kinematic coupling constrains the circumferential degree of freedom only. Linear springs to ground (SPRING1) are attached to all constrained nodes and act in the x- and z-directions. The reference node is then rotated $\mathbf{2\pi}$ about x during a static step.

Problem *xkcsort.inp* consists of a model composed of an axial arrangement of 20 shell elements. These elements are tied together using combinations of kinematic coupling constraints as well as MPCs. The constraints are defined such that kinematic coupling reference nodes appear after constraint definitions that eliminate degrees of freedom on these nodes; thus, sorting is required. The structure is clamped on one end and a concentrated axial load is applied to the other end.

Results and discussion

The tests with equivalent MPC verification problems result in identical behavior. Tests *xkcrevo.inp* and *xkcuniv.inp* result in behavior that is identical to that of the equivalent revolute and universal MPCs.

Tests *xkccirc.inp* and *xkccirc2.inp* result in motion of the constrained nodes, under action of the linear springs, as the reference node rotates. For test *xkccirc.inp* this motion remains on the local radius passing through the node at all increments. For test *xkccirc2.inp* this motion remains in the plane defined by the original configuration local radius and global z-direction as this plane rotates according to the motion prescribed at the reference node.

Test *xkcsort.inp* results in an internal sorting of MPC and kinematic coupling definitions so that a proper elimination order is achieved.

Input files

xkcbeam.inp	Equivalent to MPC type BEAM in a plane (xmpcbeam.inp).
xkcbem3.inp	Equivalent to MPC type BEAM in space (xmpcbem3.inp).
xkcrevo.inp	Test of revolute rotational constraint behavior.
xkcuniv.inp	Test of universal rotational constraint behavior.
xkccirc.inp	Local orientations defined and release of a single translational degree of freedom.
xkccirc.f	User subroutine *ORIENT* used in xkccirc.inp.
xkccirc2.inp	Local orientations defined and release of two translational degrees of freedom.
xkccirc2.f	User subroutine *ORIENT* used in xkccirc2.inp.
xkcsort.inp	Test of internal sorting of kinematic coupling definitions.

Shell-to-solid coupling constraints

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Features tested

This section provides basic verification tests for shell-to-solid coupling constraints.

Static tests with shell-to-solid coupling constraints

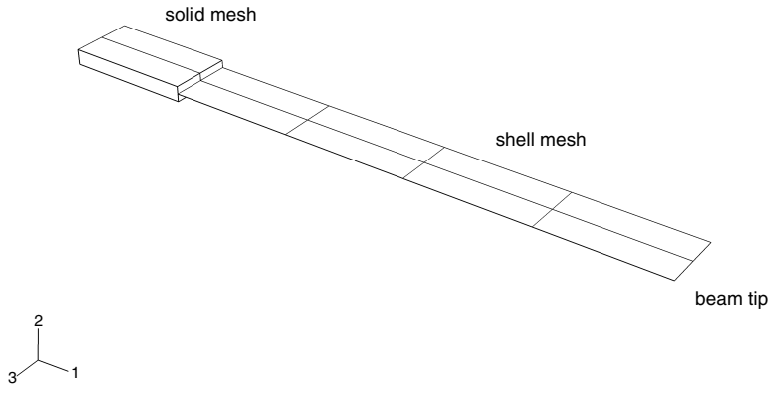
Elements tested

SC8R
S3R S4R S8R S9R5 STRI3 STRI65
C3D4 C3D8 C3D8R C3D10 C3D10HS C3D10M
C3D20R C3D27R
CSS8

Problem description

A cantilevered beam consisting of shell and continuum elements connected by shell-to-solid coupling constraints is subjected to various load conditions at the tip. The problem is analyzed with various combinations of shell and solid elements.

In addition, two input files are provided to illustrate how the shell-to-solid coupling constraints can be used to connect shell elements to continuum shell elements. In this case the continuum shell represents the solid interface.



Loading:

- Step 1: A load of F_y=–60 is applied at the tip of the beam in a linear perturbation analysis.
- Step 2: A load of F_x=60 is applied at the tip of the beam in a linear perturbation analysis.
- Step 3: A load of F_y=–60 is applied at the tip of the beam in a large-displacement analysis.
- Step 4: A load of F_x=60 is applied at the tip of the beam in a large-displacement analysis.
- Step 5: The loads that were applied in the fourth step are removed.
- Step 6: The boundary conditions are changed, and a rotation of $\pi/2$ around the z-axis is prescribed at tip of the beam.

For Abaqus/Explicit tests, the linear perturbation steps are omitted and the loading is as follows:

- Step 1: A load of F_y=–60 is applied at the tip of the beam in a large-displacement analysis.
- Step 2: A load of F_x=60 is applied at the tip of the beam in a large-displacement analysis.
- Step 3: The loads that were applied in the first two steps are removed.
- Step 4: The boundary conditions are changed, and a rotation of $\pi/2$ around the z-axis is prescribed at the tip of the beam.

Results and discussion

The results for the general cases indicate that the shell edges and solid elements are coupled appropriately.

Input files

Abaqus/Standard input files

xshell2solid_s3r_c3d4_std.inp	Shell-to-solid coupling tested between S3R shell elements and C3D4 continuum elements in a static analysis.
xshell2solid_s3r_c3d8_std.inp	Shell-to-solid coupling tested between S3R shell elements and C3D8 continuum elements in a static analysis.
xshell2solid_s3r_c3d10_std.inp	Shell-to-solid coupling tested between S3R shell elements and C3D10 continuum elements in a static analysis.
xshell2solid_s3r_c3d10hs_std.inp	Shell-to-solid coupling tested between S3R shell elements and C3D10HS continuum elements in a static analysis.
xshell2solid_s3r_c3d10m_std.inp	Shell-to-solid coupling tested between S3R shell elements and C3D10M continuum elements in a static analysis.
xshell2solid_s3r_c3d20r_std.inp	Shell-to-solid coupling tested between S3R shell elements and C3D20R continuum elements in a static analysis.
xshell2solid_s3r_c3d27r_std.inp	Shell-to-solid coupling tested between S3R shell elements and C3D27R continuum elements in a static analysis.
xshell2solid_s4r_c3d4_std.inp	Shell-to-solid coupling tested between S4R shell elements and C3D4 continuum elements in a static analysis.
xshell2solid_s4r_c3d8_std.inp	Shell-to-solid coupling tested between S4R shell elements and C3D8 continuum elements in a static analysis.

xshell2solid_s4r_c3d8_nb_std.inp	Shell-to-solid coupling tested between S4R shell elements and C3D8 continuum elements in a static analysis with a node-based surface defined on the continuum elements.
xshell2solid_s4r_c3d10_std.inp	Shell-to-solid coupling tested between S4R shell elements and C3D10 continuum elements in a static analysis.
xshell2solid_s4r_c3d10hs_std.inp	Shell-to-solid coupling tested between S4R shell elements and C3D10HS continuum elements in a static analysis.
xshell2solid_s4r_c3d10m_std.inp	Shell-to-solid coupling tested between S4R shell elements and C3D10M continuum elements in a static analysis.
xshell2solid_s4r_c3d20r_std.inp	Shell-to-solid coupling tested between S4R shell elements and C3D20R continuum elements in a static analysis.
xshell2solid_s4r_c3d20r_nb_std.inp	Shell-to-solid coupling tested between S4R shell elements and C3D20R continuum elements in a static analysis with a node-based surface defined on the continuum elements.
xshell2solid_s4r_c3d27r_std.inp	Shell-to-solid coupling tested between S4R shell elements and C3D27R continuum elements in a static analysis.
xshell2solid_s4r_sc8r_std.inp	Shell-to-solid coupling tested between S4R shell elements and SC8R continuum shell elements in a static analysis.
xshell2solid_s8r_c3d4_std.inp	Shell-to-solid coupling tested between S8R shell elements and C3D4 continuum elements in a static analysis.
xshell2solid_s8r_c3d8_std.inp	Shell-to-solid coupling tested between S8R shell elements and C3D8 continuum elements in a static analysis.
xshell2solid_s8r_c3d8_nb_std.inp	Shell-to-solid coupling tested between S8R shell elements and C3D8 continuum elements in a static analysis with a node-based surface defined on the continuum elements.
xshell2solid_s8r_c3d8_off_std.inp	Shell-to-solid coupling tested between S8R shell elements and C3D8 continuum elements in a static analysis with the OFFSET parameter used on the **SHELL SECTION* option.
xshell2solid_s8r_c3d10_std.inp	Shell-to-solid coupling tested between S8R shell elements and C3D10 continuum elements in a static analysis.
xshell2solid_s8r_c3d10hs_std.inp	Shell-to-solid coupling tested between S8R shell elements and C3D10HS continuum elements in a static analysis.
xshell2solid_s8r_c3d10m_std.inp	Shell-to-solid coupling tested between S8R shell elements and C3D10M continuum elements in a static analysis.
xshell2solid_s8r_c3d20r_std.inp	Shell-to-solid coupling tested between S8R shell elements and C3D20R continuum elements in a static analysis.
xshell2solid_s8r_c3d20r_nb_std.inp	Shell-to-solid coupling tested between S8R shell elements and C3D20R continuum elements in a static analysis with a node-based surface defined on the continuum elements.
xshell2solid_s8r_c3d27r_std.inp	Shell-to-solid coupling tested between S8R shell elements and C3D27R continuum elements in a static analysis.
xshell2solid_s9r5_c3d8_std.inp	Shell-to-solid coupling tested between S9R5 shell elements and C3D8 continuum elements in a static analysis.
xshell2solid_stri3_c3d8_std.inp	Shell-to-solid coupling tested between STRI3 shell elements and C3D8 continuum elements in a static analysis.
xshell2solid_stri65_c3d20r_std.inp	Shell-to-solid coupling tested between STRI65 shell elements and C3D20R continuum elements in a static analysis.

xshell2solid_s4r_css8_std.inp	Shell-to-solid coupling tested between S4R shell elements and CSS8 continuum solid shell elements in a static analysis.

Abaqus/Explicit input files

xshell2solid_s3r_c3d4_xpl.inp	Shell-to-solid coupling tested between S3R shell elements and C3D4 continuum elements in a static analysis.
xshell2solid_s3r_c3d8r_xpl.inp	Shell-to-solid coupling tested between S3R shell elements and C3D8R continuum elements in a static analysis.
xshell2solid_s3r_c3d10m_xpl.inp	Shell-to-solid coupling tested between S3R shell elements and C3D10M continuum elements in a static analysis.
xshell2solid_s4r_c3d4_xpl.inp	Shell-to-solid coupling tested between S4R shell elements and C3D4 continuum elements in a static analysis.
xshell2solid_s4r_c3d8r_xpl.inp	Shell-to-solid coupling tested between S4R shell elements and C3D8R continuum elements in a static analysis.
xshell2solid_s4r_c3d10m_xpl.inp	Shell-to-solid coupling tested between S4R shell elements and C3D10M continuum elements in a static analysis.
xshell2solid_s4r_sc8r_xpl.inp	Shell-to-solid coupling tested between S4R shell elements and SC8R continuum shell elements in a static analysis.

Dynamic tests with shell-to-solid coupling constraints

Elements tested

S4R S8R
C3D4 C3D8 C3D8R C3D10 C3D10HS C3D10M
C3D20R C3D27R

Problem description

A cantilevered beam consisting of shell and continuum elements connected by shell-to-solid coupling constraints is subjected to various load conditions at the tip. The problem is analyzed with various combinations of shell and solid elements.

Loading:

- Step 1: A frequency analysis is performed on the beam.
- Step 2: The beam is bent in a large-displacement analysis.
- Step 3: The beam is released, and a nonlinear dynamic springback analysis is performed.

For Abaqus/Explicit tests, the frequency analysis is omitted and the loading is as follows:

- Step 1: The beam is bent in a large-displacement analysis.
- Step 2: The beam is released, and a nonlinear dynamic springback analysis is performed.

Results and discussion

The results for the general cases indicate that the shell edges and solid elements are coupled appropriately.

Input files

Abaqus/Standard input files

xshell2solid_dyn_s4r_c3d4_std.inp	Shell-to-solid coupling tested between S4R shell elements and C3D4 continuum elements in a dynamic analysis.
xshell2solid_dyn_s4r_c3d8_std.inp	Shell-to-solid coupling tested between S4R shell elements and C3D8 continuum elements in a dynamic analysis.
xshell2solid_dyn_s4r_c3d10_std.inp	Shell-to-solid coupling tested between S4R shell elements and C3D10 continuum elements in a dynamic analysis.
xshell2solid_dyn_s4r_c3d10hs_std.inp	Shell-to-solid coupling tested between S4R shell elements and C3D10HS continuum elements in a dynamic analysis.
xshell2solid_dyn_s4r_c3d10m_std.inp	Shell-to-solid coupling tested between S4R shell elements and C3D10M continuum elements in a dynamic analysis.
xshell2solid_dyn_s4r_c3d20_std.inp	Shell-to-solid coupling tested between S4R shell elements and C3D20 continuum elements in a dynamic analysis.
xshell2solid_dyn_s4r_c3d27_std.inp	Shell-to-solid coupling tested between S4R shell elements and C3D27 continuum elements in a dynamic analysis.
xshell2solid_dyn_s8r_c3d4_std.inp	Shell-to-solid coupling tested between S8R shell elements and C3D4 continuum elements in a dynamic analysis.
xshell2solid_dyn_s8r_c3d8_std.inp	Shell-to-solid coupling tested between S8R shell elements and C3D8 continuum elements in a dynamic analysis.
xshell2solid_dyn_s8r_c3d10_std.inp	Shell-to-solid coupling tested between S8R shell elements and C3D10 continuum elements in a dynamic analysis.
xshell2solid_dyn_s8r_c3d10hs_std.inp	Shell-to-solid coupling tested between S8R shell elements and C3D10HS continuum elements in a dynamic analysis.
xshell2solid_dyn_s8r_c3d10m_std.inp	Shell-to-solid coupling tested between s8r shell elements and C3D10M continuum elements in a dynamic analysis.
xshell2solid_dyn_s8r_c3d20_std.inp	Shell-to-solid coupling tested between S8R shell elements and C3D20 continuum elements in a dynamic static analysis.
xshell2solid_dyn_s8r_c3d27_std.inp	Shell-to-solid coupling tested between S8R shell elements and C3D27 continuum elements in a dynamic analysis.

Abaqus/Explicit input files

xshell2solid_dyn_s4r_c3d4_xpl.inp	Shell-to-solid coupling tested between S4R shell elements and C3D4 continuum elements in a dynamic analysis.
xshell2solid_dyn_s4r_c3d8r_xpl.inp	Shell-to-solid coupling tested between S3R shell elements and C3D8R continuum elements in a dynamic analysis.
xshell2solid_dyn_s4r_c3d10m_xpl.inp	Shell-to-solid coupling tested between S4R shell elements and C3D10M continuum elements in a dynamic analysis.

Free vibration of a cantilevered thin square with shell-to-solid coupling constraints

Elements tested

S8R STRI65
C3D10 C3D10HS C3D20R

Problem description

A free vibration analysis is carried out for a cantilevered thin square plate (see *Figure 1*). The outside section of the plate is modeled with shell elements, and the middle section of the plate is modeled with continuum elements coupled to the shell elements using shell-to-solid coupling constraints. The first six modes are extracted. The problem is analyzed with various combinations of shell and solid elements. These tests verify the ability of coupling constraints to model the shell-to-solid coupling accurately with an interface that includes corners. The free surface generation capability for both the shell and solid elements is also tested.

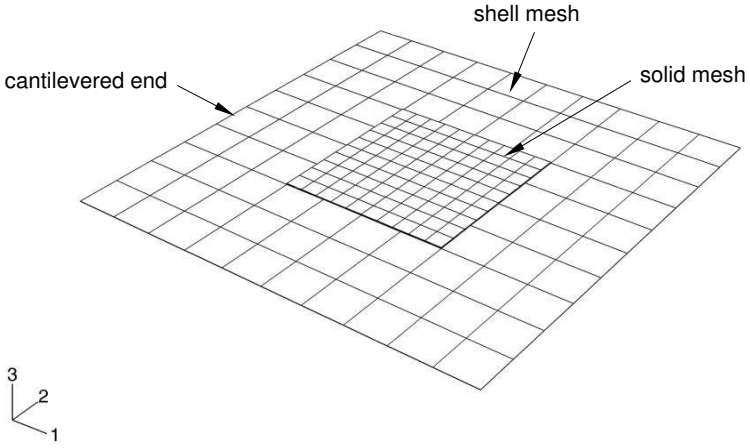


Figure 1: Cantilevered thin square plate.

Results and discussion

The natural frequencies and mode shapes compare well to the reference NAFEMS solution. The NAFEMS solution is taken from the National Agency for Finite Element Methods and Standards (U.K.): Test FV16 from NAFEMS publication TNSB, Rev. 3, “The Standard NAFEMS Benchmarks,” October 1990.

1. Test1 — S8R shell elements and C3D10 continuum elements (with and without free surface generation).
2. Test2 — S8R shell elements and C3D10HS continuum elements (with and without free surface generation).
3. Test3 — S8R shell elements and C3D20 continuum elements (with and without free surface generation).
4. Test4 — STRI65 shell elements and C3D10 continuum elements.
5. Test5 — STRI65 shell elements and C3D10HS continuum elements.
6. Test6 — STRI65 shell elements and C3D20 continuum elements.

	Mode					
	1	2	3	4	5	6
NAFEMS	0.421	1.029	2.582	3.306	3.753	6.555
Test 1	0.434	1.024	2.861	3.642	3.873	6.745
Test 2	0.434	1.024	2.861	3.642	3.873	6.745
Test 3	0.429	1.023	2.750	3.484	3.809	6.641
Test 4	0.434	1.024	2.875	3.628	3.866	6.727
Test 5	0.434	1.024	2.875	3.628	3.866	6.727
Test 6	0.430	1.024	2.782	3.496	3.811	6.648

Input files

Abaqus/Standard input files

xshell2solidvib_c3d10_s8r.inp	Free vibration analysis of a cantilevered thin square plate with S8R shell elements and C3D10 continuum elements coupled together using shell-to-solid coupling constraints.
xshell2solidvib_c3d10_s8r_free.inp	Free vibration analysis of a cantilevered thin square plate with S8R shell elements and C3D10 continuum elements coupled together using shell-to-solid coupling constraints. Free surface generation is used for both solid and shell surfaces.
xshell2solidvib_c3d10_stri65.inp	Free vibration analysis of a cantilevered thin square plate with STRI65 shell elements and C3D10 continuum elements coupled together using shell-to-solid coupling constraints.
xshell2solidvib_c3d10hs_s8r.inp	Free vibration analysis of a cantilevered thin square plate with S8R shell elements and C3D10HS continuum elements coupled together using shell-to-solid coupling constraints.
xshell2solidvib_c3d10hs_s8r_free.inp	Free vibration analysis of a cantilevered thin square plate with S8R shell elements and C3D10HS continuum elements coupled together using shell-to-solid coupling constraints. Free surface generation is used for both solid and shell surfaces.
xshell2solidvib_c3d10hs_stri65.inp	Free vibration analysis of a cantilevered thin square plate with STRI65 shell elements and C3D10HS continuum elements coupled together using shell-to-solid coupling constraints.
xshell2solidvib_c3d20_s8r.inp	Free vibration analysis of a cantilevered thin square plate with S8R shell elements and C3D20 continuum elements coupled together using shell-to-solid coupling constraints.
xshell2solidvib_c3d20_s8r_free.inp	Free vibration analysis of a cantilevered thin square plate with S8R shell elements and C3D20 continuum elements coupled together using shell-to-solid coupling constraints. Free surface generation is used for both solid and shell surfaces.
xshell2solidvib_c3d20_stri65.inp	Free vibration analysis of a cantilevered thin square plate with STRI65 shell elements and C3D20 continuum elements coupled together using shell-to-solid coupling constraints.

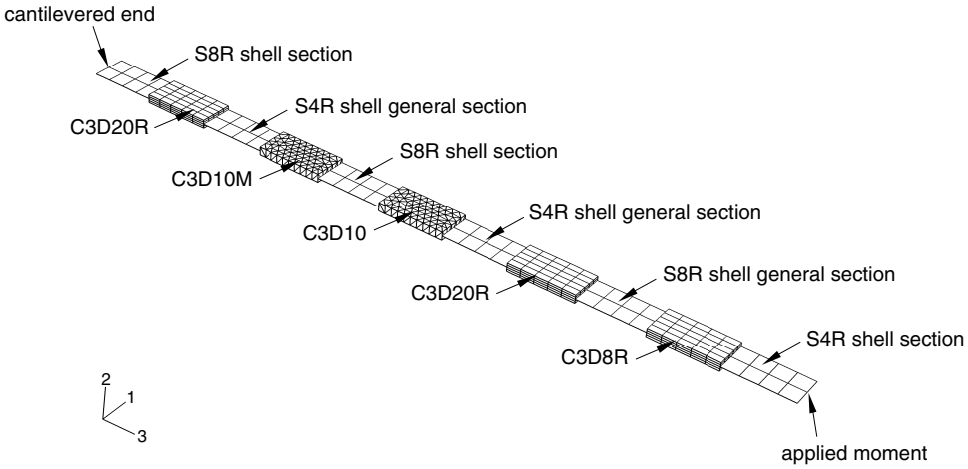
Static test of a built-up beam with shell-to-solid coupling constraints

Elements tested

S4R S8R
C3D8R C3D10 C3D10M
C3D20R

Problem description

The pure bending of a cantilevered beam is modeled with an alternating mesh of shell and continuum elements. Ten separate shell-to-solid interfaces are modeled in this example. The beam is 22 in long, 1 in wide, and 0.25 in thick. The material is linear elastic with a Young's modulus of 30×10^6 psi and Poisson's ratio of 0.3. The reference tip displacement solution from classical linear elasticity for a moment of 400 lb-in is −2.4 in.



Loading:

- Step 1: A moment of M_x = 400 lb-in is applied at the tip of the beam in a linear perturbation analysis.
- Step 2: A moment of M_x = 400 lb-in is applied at the tip of the beam using NLGEOM for large-displacement analysis.

Results and discussion

The results for the general cases indicate that the shell edges and solid elements are coupled appropriately. The computed tip displacements for the linear perturbation and nonlinear analyses are −2.49 in and −2.48 in, respectively.

Input files

xshell2solid_builtupbeam.inp Shell-to-solid coupling tested for built-up beam in a static analysis.

Explicit dynamic test of a built-up beam with shell-to-solid coupling constraints

Elements tested

S4R
C3D8R C3D4 C3D10M

Problem description

The bending of a cantilevered beam is modeled with an alternating mesh of shell and continuum elements. Ten separate shell-to-solid interfaces are modeled in this example. The beam is 22 in long, 1 in wide, and 0.25 in thick. The material is linear elastic with a Young's modulus of 30×10^6 psi and $\mu = .3$. The beam is subjected to a tip displacement of −2.4 in.

Loading:

- Step 1: A displacement of $u_y = -2.4$ in is applied at the tip of the beam.

Results and discussion

The results for the general cases indicate that the shell edges and solid elements are coupled appropriately.

Input files

xshell2solid_builtupbeam_xpl.inp	Shell-to-solid coupling tested for built-up beam in a explicit dynamic analysis.

Coupling of acoustic and structural elements

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

ASI1 ASI2 ASI2A ASI3 ASI3A ASI4 ASI8

Problem description

The model consists of a column of fluid 100 units high with a cross-sectional area of 400. The fluid column is modeled with five acoustic elements; five high and one in the cross-section. The top of the column has a zero pressure boundary condition applied, thus representing a free surface. The base of the column is connected to structural degrees of freedom via an acoustic-structural interface element.

A dynamic analysis is performed during which a sinusoidal acceleration is applied to the base of the fluid column via the interface element. The pressure distribution throughout the fluid column is determined after one unit of dynamic time has elapsed.

Material:

Bulk modulus	2×10^9
Density	1000.0

Acoustic link element model:

Link length	20.0
Link area	400.0

Two-dimensional acoustic element model:

Acoustic element size	20.0×20.0
Element thickness	20.0

Axisymmetric acoustic element model:

Acoustic element size	11.28379×20.0

Three-dimensional acoustic element model:

Acoustic element dimensions	$20.0 \times 20.0 \times 20.0$
test	test

Results and discussion

a_z at node 1 is the same for all tests.

Input files

ei11aca1.inp	ASI1, AC1D2 elements.
ei11aca2.inp	ASI1, AC1D3 elements.
ei22aca1.inp	ASI2, AC2D4 elements.
eia2aca1.inp	ASI2A, ACAX4 elements.
ei23aca2.inp	ASI3, AC2D8 elements.
eia3aca2.inp	ASI3A, ACAX8 elements.
ei34aca1.inp	ASI4, AC3D8 elements.
ei38aca2.inp	ASI8, AC3D20 elements.

Mesh-independent spot welds

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Features tested

This section provides basic verification tests for the mesh-independent spot weld and mesh-independent spot-weld properties procedures.

Spot-welded plates subject to pressure and shear loading

Elements tested

S4 S4R

Problem description

Rigid spot welds are defined between combinations of two or more plates comprised of three-dimensional shell elements. The spot weld options are used to test the various ways in which the user can define mesh-independent spot welds. The three ways in which the user can define the spot-welded surfaces are verified: the user does not specify any surface, the user specifies a single surface, or the user specifically lists the surfaces to be spot welded. Limiting the surface facets considered for spot welding is verified, along with controlling the distributing coupling definitions generated by the spot welds. In addition, user-specified projection directions are tested. Structural coupling is also tested for many of the test combinations above.

Each combination is subjected to the same loading conditions. In the Abaqus/Standard analyses the top plate is loaded with a uniform pressure. In the Abaqus/Explicit analyses the top and bottom plates in each combination are subjected to displacements of u_x=.1 and u_x=−.1, respectively, along the plate edges parallel to the y-axis.

Results and discussion

The results for each combination indicate that the surfaces are spot welded appropriately.

Input files

Abaqus/Standard input files

fastener_projdir_s4.inp	Tests user-specified projection directions for spot welds with a single user-specified surface.
fastener_search_s4.inp	Tests the RADIUS OF INFLUENCE, SEARCH RADIUS, and NUMBER OF LAYERS parameters for spot welds with no user-specified surface.
fastener_unsort_weight_s4.inp	Tests the UNSORTED and WEIGHTING METHOD parameters with multiple user-specified surfaces.

Abaqus/Explicit input files

xfastener_xpl_beammpcs.inp	Tests the various methods for defining mesh-independent spot welds using BEAM-type MPCs.
xfastener_xpl_beammpcs_struct.inp	Same as above only using structural coupling.
xfastener_xpl_connectors.inp	Tests the various methods for defining mesh-independent spot welds using both user-defined and internally generated connector elements.
xfastener_xpl_connectors_struct.inp	Same as above only using structural coupling.
xfastener_xpl_beammpcs_s4.inp	Tests the various methods for defining mesh-independent spot welds using BEAM-type MPCs with S4 elements.
xfastener_xpl_beammpcs_struct_s4.inp	Same as above only using structural coupling.
xfastener_xpl_connectors_s4.inp	Tests the various methods for defining mesh-independent spot welds using both user-defined and internally generated connector elements.

Multi-layer spot welds between surfaces defined on various element types

Elements tested

S3 S4 S8R STRI65

Problem description

Various combinations of plates are spot welded to the faces of a bi-unit cube. These tests verify the ability of Abaqus to accurately spot weld meshes of different element types. These tests also verify several features of the mesh-independent fastener and mesh-independent fastener properties procedures including user-specified and free surface options, default and user-specified orientations and projection directions, multiple interactions, fastener property and reference node options, and fully constrained and released rotation constraints.

Results and discussion

The results indicate that the fastener options tested are modeled correctly.

Input files

fastener_multilay_lin_std.inp	Plates spot welded to a cube with user-specified surfaces and orientations; static linear perturbation tests including multiple load cases.
fastener_multilay_lin_conn_ std.inp	Plates spot welded to a cube with user-specified surfaces and orientations; static linear perturbation tests including multiple load cases. BEAM connector elements are used instead of BEAM-type MPCs.
fastener_multilay_lin_r1_std.inp	Plates spot welded to a cube with user-specified surfaces and orientations; rotation constraint in spot welds released in the local 3-direction; static linear perturbation tests including multiple load cases.
fastener_multilay_lin_r3_std.inp	Plates spot welded to a cube with user-specified surfaces and orientations; all rotation constraints in spot welds released; static linear perturbation tests including multiple load cases.
fastener_multilay_free_lin_std.inp	Plates spot welded to a cube with free and user-specified surfaces and orientations and user-specified projection directions; static linear perturbation tests including multiple load cases.
fastener_s4_multilay_std.inp	Plates spot welded to a cube with user-specified surfaces and orientations; static linear perturbation and geometrically nonlinear tests; S4 elements.

Single-layer spot welds between surfaces defined on various element types with varying mesh densities

Elements tested

S3 S4 S4R S8R
C3D4 C3D8R C3D10M C3D20R
R3D3 R3D4

Problem description

Individual plates are spot welded to the faces of a cube. These tests verify the mesh-independent fastener procedure in both perturbation and geometrically nonlinear analyses, including restart. These tests also verify fasteners on meshes of varying density. In addition, structural coupling is also tested.

Results and discussion

The results indicate that the fastener options tested are modeled correctly.

Input files

Abaqus/Standard input files

fastener_s4_std.inp	Plates spot welded to a cube with user-specified surfaces and orientations; static linear perturbation, frequency extraction, direct and mode-based steady-state dynamic, and geometrically nonlinear tests; S4 elements.
fastener_s4_struct_std.inp	Plates spot welded to a cube using structural coupling with user-specified surfaces and orientations; static linear perturbation, frequency extraction, direct and mode-based steady-state dynamic, and geometrically nonlinear tests; S4 elements.
fastener_s4_std_res.inp	Restart analysis of fastener_s4_std.inp; S4 elements.
fastener_s8r_std.inp	Plates spot welded to a cube with user-specified surfaces and orientations; static linear perturbation, frequency extraction, direct and mode-based steady-state dynamic, and geometrically nonlinear tests; S4 elements.
fastener_s3_c3d4_std.inp	Plate spot welded to a cube with user-specified surfaces; single static step; S3 and C3D4 elements with varying mesh density.
fastener_s3_c3d8r_std.inp	Plate spot welded to a cube with user-specified surfaces; single static step; S3 and C3D8R elements with varying mesh density.
fastener_s3_c3d10m_std.inp	Plate spot welded to a cube with user-specified surfaces; single static step; S3 and C3D10M elements with varying mesh density.
fastener_s3_c3d20r_std.inp	Plate spot welded to a cube with user-specified surfaces; single static step; S3 and C3D20R elements with varying mesh density.
fastener_s4r_c3d4_std.inp	Plate spot welded to a cube with user-specified surfaces; single static step; S4R and C3D4 elements with varying mesh density.
fastener_s4r_c3d8r_std.inp	Plate spot welded to a cube with user-specified surfaces; single static step; S4R and C3D8R elements with varying mesh density.
fastener_s4r_c3d10m_std.inp	Plate spot welded to a cube with user-specified surfaces; single static step; S4R and C3D10M elements with varying mesh density.

fastener_s4r_c3d20r_std.inp	Plate spot welded to a cube with user-specified surfaces; single static step; S4R and C3D20R elements with varying mesh density.
fastener_s8r_c3d4_std.inp	Plate spot welded to a cube with user-specified surfaces; single static step; S8R and C3D4 elements with varying mesh density.
fastener_s8r_c3d8r_std.inp	Plate spot welded to a cube with user-specified surfaces; single static step; S8R and C3D8R elements with varying mesh density.
fastener_s8r_c3d8r_struct_std.inp	Plate spot welded to a cube using structural coupling with user-specified surfaces; single static step; S8R and C3D8R elements with varying mesh density.
fastener_s8r_c3d10m_std.inp	Plate spot welded to a cube with user-specified surfaces; single static step; S8R and C3D10M elements with varying mesh density.
fastener_s8r_c3d20r_std.inp	Plate spot welded to a cube with user-specified surfaces; single static step; S8R and C3D20R elements with varying mesh density.
fastener_r3d3_c3d4_std.inp	Plate spot welded to a cube with user-specified surfaces; single static step; R3D3 and C3D4 elements with varying mesh density.
fastener_r3d4_c3d10m_std.inp	Plate spot welded to a cube with user-specified surfaces; single static step; R3D4 and C3D10M elements with varying mesh density.

Abaqus/Explicit input files

xfastener_xpl_c3d10m_m3d4r.inp	Surface comprised of M3D4R membrane elements spot welded to surface comprised of C3D10M continuum elements.
xfastener_xpl_c3d10m_s4r_struct.inp	Surface comprised of S4R membrane elements spot welded to surface comprised of C3D10M continuum elements using structural coupling.
xfastener_xpl_c3d4_r3d4.inp	Surface comprised of R3D4 rigid elements spot welded to surface comprised of C3D4 continuum elements.
xfastener_xpl_c3d8r_s3r.inp	Surface comprised of S3R shell elements spot welded to surface comprised of C3D8R continuum elements.
xfastener_xpl_c3d8r_s3r_struct.inp	Same as above only using structural coupling.

Large deformation of a spot-welded beam

Elements tested

S4

Problem description

Two beams are spot welded together and subjected to various geometrically nonlinear deformations.

Results and discussion

The results indicate that the spot welds are modeled correctly.

Input files

fastenedbeam_s4_s4.inp	Spot-welded beams, S4 elements.
fastenedbeam_s4_s4_struct.inp	Spot-welded beams using structural coupling, S4 elements.
fastenedbeam_s4_s4_po.inp	Post output analysis of fastenedbeam_s4_s4.inp.

fastenedbeam_s4_s4_struct_lin.inp	Spot-welded beams using structural coupling, S4 elements. Geometrically linear analysis.
fastenedbeam_s4_s4_struct_pert.inp	Spot-welded beams using structural coupling, S4 elements. Perturbation analysis.

Spot welds used in various analysis techniques

Elements tested

C3D20R S4R

Problem description

The following examples verify that spot welds work with the following analysis techniques: mesh removal and activation, submodeling, and substructures.

Results and discussion

The results of these tests indicate that spot welds are modeled correctly for these analysis techniques.

Input files

fastener_mdlc_s4r_c3d20r.inp	Geometrically nonlinear static and dynamic analyses (including element removal) of a spot-welded model consisting of S4 and C3D20R elements.
fastener_struct_mdlc_s4r_c3d20r.inp	Geometrically nonlinear static and dynamic analyses (including element removal) of a spot-welded model consisting of S4 and C3D20R elements.
fastener_s4r_global.inp	Static analysis of a global model with spot welds, S4R elements.
fastener_s4r_submodel.inp	Static submodel analysis of fastener_s4r_global.inp with spot welds, S4R elements.
fastener_substr_gen.inp	Substructure generation file of a spot-welded model using S4R and C3D20R elements.
fastener_substr.inp	Substructure analysis of a spot-welded model using S4R and C3D8R elements; uses fastener_substr_gen.inp for substructure generation.

Spot weld surfaces forming T-intersections

Elements tested

S4R

Problem description

The following example verifies the ability of Abaqus to accurately create fasteners between plates that are oriented perpendicular to each other; i.e., forming a T-intersection. Various combinations of plates that are perpendicular to each other, as well as plates that butt against each other, are used to verify that fasteners are formed correctly for all these cases.

Results and discussion

The results of these tests indicate that Abaqus correctly fastens plates forming T-intersections.

Input files

fastener_facetoedge_xpl.inp	Fasten surfaces forming T-intersection.

Linear dynamics

Elements tested

C3D8 S4

Problem description

A single shell element is spot welded to a single brick element. This model is analyzed using various linear dynamic procedures: steady-state dynamics (mode-based, direct, subspace), modal dynamics, random response, and spectrum response. The results of the spot-welded model are compared to similar models using connectors, beams, and distributing coupling elements. Specification of the additional mass that will be distributed to the fastener nodes is also tested.

Results and discussion

Comparison of the spot weld model results to the results from a beam model indicates that the spot welds are modeled correctly.

Input files

fastener_lindyn.inp	Spot-welded model using S4 and C3D8 elements.
fastener_lindyn_connect.inp	Spot-welded model using S4 and C3D8 elements. BEAM connector elements are used instead of BEAM type MPCs.
fastener_lindyn_beam.inp	Spot-welded model using B31 and C3D8 elements.
fastener_lindyn_mass.inp	Spot-welded model using S4 and C3D8 elements and the MASS parameter.

Compounded local coordinate systems

Elements tested

C3D8 S4

Problem description

If a connector element is used to model a fastener, the local coordinate system defined on the connector section ($\mathbf{T}_{connector}$) operates on the local coordinate system for the fastener ($\mathbf{T}_{fastener}$) to determine the final local coordinate system of the connector element ($\mathbf{T}_{connectorfinal}$). In other words,

$$\mathbf{T}_{connectorfinal} = \mathbf{T}_{connector} * \mathbf{T}_{fastener}.$$

In the above equations $\mathbf{T}_{connector}$ and $\mathbf{T}_{fastener}$ are assumed to be orthogonal rotation matrices with the local 1-, 2-, and 3-directions being the first, second and, third rows, respectively. The local coordinate system for a connector element modeling a fastener should be specified with respect to the local coordinate system of the fastener.

In the first example six flat shell structures are fastened independently to the six sides of a single brick element.

HINGE connectors have been used with their local 1-directions set to be $(0., 0., 1.)$; i.e., the local 3-direction of the fasteners. When compounded with the local coordinate system for the fasteners, the local 1-direction for the connector is normal to the surface. Thus, the shell structures are free to rotate about the surface normals.

In the second example six flat shell structures are again fastened independently to the six sides of a single brick element. TRANSLATOR connectors have been used. The local 1-directions (which are the slide direction for this type of connector: see *Connector Element Library*) for the connectors have been set to the local 1-directions of the fasteners. For the four fasteners on the side, the local 1-directions coincide with the global 2-directions. For the two fasteners on the top and the bottom, the local 1-directions coincide with the global 3-directions. Thus, the fastened shell structures on the sides are free to translate in the global 2-directions, while the fastened shell structures on the top and bottom are free to translate in the global 3-direction.

Results and discussion

The results indicate that the local coordinate systems of the HINGE and TRANSLATOR connectors are modeled correctly.

Input files

fastener_connect_hinge.inp	Six flat shell structures fastened to a cube with user-specified surfaces; single static step; S4 and C3D8 elements. HINGE connector elements are used.
fastener_connect_translator.inp	Six flat shell structures fastened to a cube with user-specified surfaces; single static step; S4 and C3D8 elements. TRANSLATOR connector elements are used.

Embedded element technique

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D8 C3D8R C3D20
CAX4 CAX4R CAX8
CPE4 CPE4R CPE8
MAX1 MAX2 M3D4 M3D4R M3D8
SFMAX1 SFMAX2 SFM3D4 SFM3D4R SFM3D8
T2D2 T2D3 T3D2 T3D3

Features tested

Various types of elements that lie embedded in different types of host elements are used to constrain the embedded nodes to the appropriate host elements.

Problem description

The models using continuum elements as host elements consist of three host elements and, in most cases, two embedded elements of different types. All the nodes at one end (x=1) are constrained in all degrees of freedom. Concentrated loads are applied in the negative y-direction to the nodes at the other end (x=10).

Results and discussion

The results obtained using the embedded element technique are the same as those obtained using an equivalent MPC model.

Input files

Abaqus/Standard input files

xembedele2d1_std.inp	Static step; a 2D first-order truss element and a 2D first-order solid element lie embedded in three 2D first-order solid elements.
xembedele2d2_std.inp	Static step; a 2D second-order truss element and a 2D second-order solid element lie embedded in three 2D second-order solid elements.
xembedelecax1_std.inp	Static step; an axisymmetric first-order membrane element with rebar and an axisymmetric first-order solid element lie embedded in three axisymmetric first-order solid elements.
xembedelecax2_std.inp	Static step; an axisymmetric second-order membrane element with rebar and an axisymmetric second-order solid element lie embedded in three axisymmetric second-order solid elements.

xembedelecax3_std.inp	Static step; an axisymmetric first-order surface element with rebar and an axisymmetric first-order solid element lie embedded in three axisymmetric first-order solid elements.
xembedelecax4_std.inp	Static step; an axisymmetric second-order surface element with rebar and an axisymmetric second-order solid element lie embedded in three axisymmetric second-order solid elements.
xembedele3d1_std.inp	Static step; a 3D first-order membrane element with rebar and a 3D first-order solid element lie embedded in three 3D first-order solid elements.
xembedele3d2_std.inp	Static step; a 3D second-order membrane element with rebar and a 3D second-order solid element lie embedded in three 3D second-order solid elements.
xembedele3d3_std.inp	Static step followed by frequency, steady-state dynamics, modal dynamics, response spectrum, random response, and dynamics steps; a 3D first-order truss element and a 3D first-order membrane element with rebar lie embedded in three 3D first-order solid elements.
xembedele3d4_std.inp	Static step; a 3D second-order truss element and a 3D second-order membrane element with rebar lie embedded in three 3D second-order solid elements.
xembedele3d5_std.inp	Static step; a 3D first-order surface element with rebar and a 3D first-order solid element lie embedded in three 3D first-order solid elements.
xembedele3d6_std.inp	Static step; a 3D second-order surface element with rebar and a 3D second-order solid element lie embedded in three 3D second-order solid elements.
xembedele3d7_std.inp	Static step followed by frequency, steady-state dynamics, modal dynamics, response spectrum, random response, and dynamics steps; a 3D first-order truss element and a 3D first-order surface element with rebar lie embedded in three 3D first-order solid elements.
xembedele3d8_std.inp	Static step; a 3D second-order truss element and a 3D second-order surface element with rebar lie embedded in three 3D second-order solid elements.
xembedele3d9_std.inp	Static step; a 3D first-order surface element with rebar and a 3D first-order solid element lie embedded in three 3D first-order solid elements. The **MODEL CHANGE* capability is applied to remove and add surface elements during the static analysis.
xembedele3d10_std.inp	Static step; first-order cylindrical surface elements with rebar lie embedded in cylindrical solid elements; 180° segment modeled.
xembedele3d11_std.inp	Static step; second-order cylindrical surface elements with rebar lie embedded in cylindrical solid elements;180° segment modeled.
xembedele3d12_std.inp	Static step; second-order cylindrical surface elements with rebar lie embedded in triangular cylindrical solid elements; 180° segment modeled.
xembedele3d13_std.inp	Static step; second-order cylindrical surface elements with rebar lie embedded in second-order cylindrical solid elements.
xembedele3d14_std.inp	Static step; shell elements with rebar and beam elements lie embedded in three 3D first-order solid elements.

Abaqus/Explicit input files

xembedele2d1_xpl.inp	A 2D first-order truss element and a 2D first-order solid element lie embedded in three 2D first-order solid elements.
xembedelecax1_xpl.inp	An axisymmetric first-order solid element lies embedded in three axisymmetric first-order solid elements.
xembedele3d1_xpl.inp	A 3D first-order membrane element with rebar and a 3D first-order solid element lie embedded in three 3D first-order solid elements.

xembedele3d1_xpl_c3d8.inp	A 3D first-order membrane element with rebar and a 3D first-order solid element lie embedded in three 3D first-order solid elements.
xembedele3d3_xpl.inp	A 3D first-order truss element and a 3D first-order membrane element with rebar lie embedded in three 3D first-order solid elements.
xembedele3d4_xpl.inp	A 3D first-order surface element with rebar and a 3D first-order solid element lie embedded in three 3D first-order solid elements.
xembedele3d5_xpl.inp	Shell elements with rebar and beam elements lie embedded in three 3D first-order solid elements.

Figures

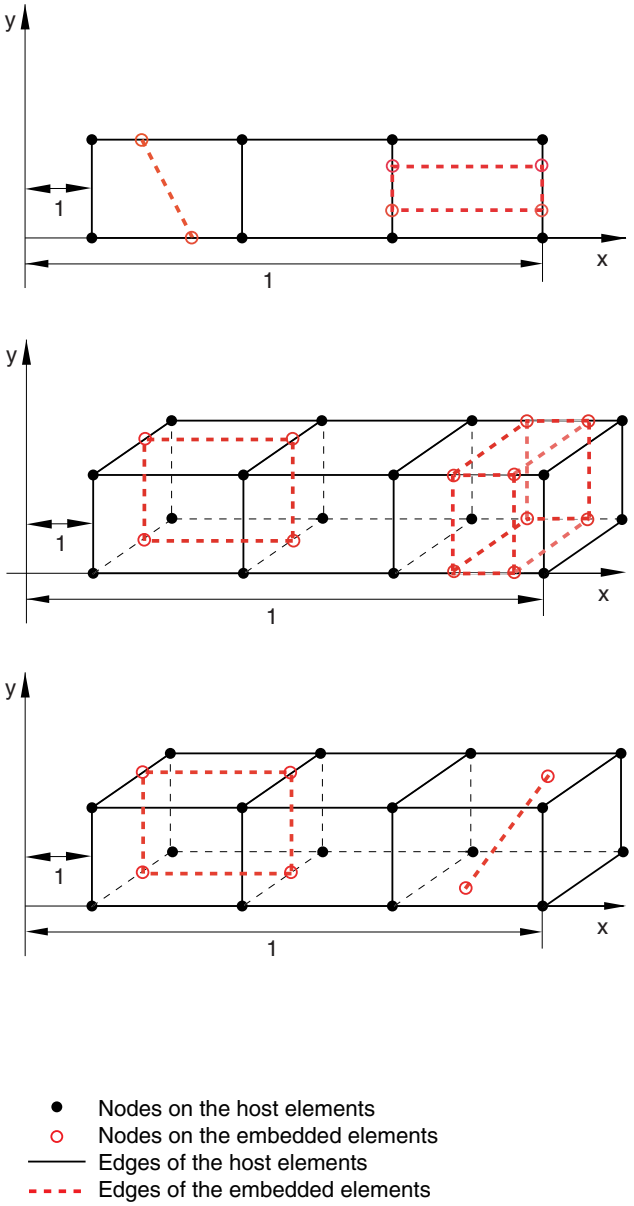


Figure 1: Elements lie embedded in host elements.

Miscellaneous

This section defines the basic tests that are used to provide evidence that the implementation of the miscellaneous constraint options in Abaqus produces the expected results.

In this section:

- *Reset overconstraint checking controls*
- *Thermal expansion of constraints*

Reset overconstraint checking controls

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Features tested

There are features in Abaqus that, when used in combination, may overconstrain a model. Several of these combinations are detected and resolved by Abaqus automatically, while others are only identified and warning or error messages are issued. By default, overconstraint checking is performed.

Intersecting surface-based tie constraints

Problem description

A surface-based tie constraint joins two surfaces by eliminating the nodes on the secondary surface with multi-point constraints. Multiple surface-based tie constraint definitions may intersect. At these intersections the secondary nodes are involved in an overconstraint. Only one surface-based tie constraint is needed to eliminate a secondary node. Additional surface-based tie constraint definitions are not needed.

In these tests intersecting surface-based tie constraint definitions are used such that one or more secondary nodes are included in more than one surface-based tie constraint pair. Only one surface-based tie constraint should be enforced at any secondary node.

Results and discussion

These tests verify that Abaqus properly removes the consistent overconstraints. Warning or error messages in the data file confirm that all overconstraints are removed or identified.

Input files

overcon_tie_tie_3d.inp	C3D8 element test for three-way **TIE* intersection.
overcon_tie_tie_axi.inp	CAX4 element test.
overcon_tie_tie_beam.inp	B21 element test, T-junction.
overcon_tie_tie_shell.inp	S4 element test, T-junction.
overcon_tie_tie_quad.inp	S8R and B32element test with three-way **TIE* intersection.

Surface-based tie and rigid body constraints

Problem description

A rigid body constraint eliminates all the degrees of freedom at the nodes of the rigid body in favor of the degrees of freedom at the reference node. Therefore, any surface-based tie constraints used to tie surfaces inside a single rigid body or between rigid bodies is a consistent overconstraint. In this case the surface-based tie constraint is ignored. Similarly, if the surface-based tie constraint is used to tie a rigid surface to a deformable surface and the surface on the rigid body is the secondary surface, a consistent overconstraint exists for the tie nodes on the rigid body. If possible, Abaqus reverses the main/secondary pair.

In these tests the surface-based tie constraints tie surfaces within a rigid body, between rigid bodies, or between a secondary rigid body and a main deformable body.

Results and discussion

These tests verify that Abaqus properly removes the consistent overconstraint. Warning messages in the data file confirm that the consistent overconstraints are removed.

Input files

overcon_rigbm_tie_rigbm.inp	**TIE* within a rigid body in two dimensions.
overcon_rigmisc_tie.inp	**TIE* within a rigid body in three dimensions.
overcon_rig_tie_rig_2d.inp	**TIE* between rigid bodies in two dimensions.
overcon_rig_tie_def_2d.inp	**TIE* between a secondary rigid surface and a deformable main surface in two dimensions.
overcon_rig_tie_rig_3d.inp	**TIE* between rigid bodies in three dimensions.
overcon_rig_tie_def_3d.inp	**TIE* between a secondary rigid surface and a deformable main surface in three dimensions.

Intersecting rigid body constraints

Problem description

If the rigid body constraint refers to nodes or elements that are already part of a rigid body, the common nodes will be involved in a consistent overconstraint.

In these tests rigid body constraints are used to create a single rigid body from other individual rigid bodies or to define a rigid body that includes a part of another rigid body.

Results and discussion

These tests verify that Abaqus properly removes the consistent overconstraints. Warning messages in the data file confirm that the consistent overconstraints are removed.

Input files

overcon_intersect_rig_2d.inp	**RIGID BODY* intersection in two dimensions.
overcon_intersect_rig_3d.inp	**RIGID BODY* intersection in three dimensions.

Surface-based tie constraints and boundary conditions

Problem description

A surface-based tie constraint eliminates the degrees of freedom at the secondary nodes using multi-point constraints. If a boundary condition is imposed on the secondary node, an overconstraint results.

In these tests two surfaces are tied and boundary conditions are assigned to the secondary nodes such that a consistent overconstraint is created.

Results and discussion

These tests verify that Abaqus properly removes the consistent overconstraints. Warning messages in the data file confirm that the consistent overconstraints are removed.

Input files

overcon_tie_boundary_2d.inp	**TIE* and **BOUNDARY* in two dimensions.
overcon_tie_boundary_3d.inp	**TIE* and **BOUNDARY* in three dimensions.

Rigid body constraints and boundary conditions

Problem description

Rigid body constraints create a rigid body that eliminates the degrees of freedom at all the nodes on the rigid body in favor of the degrees of freedom at the reference node. If a boundary condition is defined at one of the eliminated nodes, an overconstraint results.

In these tests a rigid body is defined and boundary conditions are assigned to the eliminated nodes on the rigid body such that a consistent overconstraint is created.

Results and discussion

These tests verify that Abaqus properly removes the consistent overconstraints. Warning messages in the data file confirm that the consistent overconstraints are removed.

Input files

overcon_rb_boundary_2d.inp	**RIGID BODY*and **BOUNDARY* in two dimensions.
overcon_rb_boundary_3d.inp	**RIGID BODY* and **BOUNDARY* in three dimensions.

Connector elements and rigid body constraints

Problem description

If connector elements are used to connect nodes within a rigid body, a consistent overconstraint is introduced since the nodes at both ends of the connector element already have a rigid constraint. In this case the connector element should be removed. If multiple connector elements are used between rigid bodies, all kinematic constraints beyond three translational constraints and three rotational constraints (in three dimensions) or two translational constraints and one rotational constraint (in two dimensions) are overconstraints. In the case when the connector elements produce a consistent overconstraint between the two rigid bodies, all the connector elements are removed and a connector element of type BEAM is attached between the two rigid body reference nodes.

In these tests connector elements are connected either between nodes within a rigid body or between nodes on different rigid bodies.

Results and discussion

These tests verify that Abaqus properly removes the consistent overconstraints. Warning messages in the data file confirm that the consistent overconstraints are removed.

Input files

overcon_conn_between_rig_2d.inp	Connector elements between rigid bodies in two dimensions.
overcon_conn_between_rig_3d.inp	Connector elements between rigid bodies in three dimensions.
overcon_conn_inside_rig_2d.inp	Connector elements inside a rigid body in two dimensions.
overcon_conn_inside_rig_3d.inp	Connector elements inside a rigid body in three dimensions.

Coupling and rigid body constraints

Problem description

Rigid body constraints eliminate all the degrees of freedom at the nodes belonging to the rigid body. If coupling constraints are also used, an overconstraint may occur. Abaqus/Standard will automatically eliminate the unnecessary coupling constraints.

Results and discussion

These tests verify that Abaqus properly removes the consistent overconstraints. Warning or error messages in the data file confirm that these overconstraints are removed or identified.

Input files

overcon_rb_coup.inp	The **RIGID BODY* and **COUPLING* options with the **DISTRIBUTING* option.
overcon_rb_kc.inp	The **RIGID BODY* and **COUPLING* options with the **KINEMATIC* option.

Contact interactions and surface-based tie constraints

Problem description

A surface-based tie constraint eliminates the degrees of freedom at the secondary node through multi-point constraints. If the tied surfaces intersect a surface where a contact interaction is defined (normal contact with or without Lagrange friction), the contact interactions at the secondary node are overconstraints.

In these tests surface-based tie constraints intersect surfaces with contact interactions.

Results and discussion

These tests verify that Abaqus properly removes the consistent overconstraints. Warning messages in the data file confirm that the consistent overconstraints are removed.

Input files

overcon_tie_contact_2d.inp	Contact and **TIE* in two dimensions.
overcon_tie_contact_3d.inp	Contact and **TIE* in three dimensions.

Contact interactions and boundary conditions

Problem description

Contact interactions and prescribed boundary conditions may lead to overconstraints if either normal contact with the default “hard contact” formulation or Lagrange frictional contact is used.

In these tests hard contact or Lagrange friction is defined and boundary conditions are applied to contact secondary nodes.

Results and discussion

These tests verify that Abaqus properly removes the consistent overconstraints. Warning messages in the data file confirm that the consistent overconstraints are removed.

Input files

overcon_bc_contact_2d.inp	Normal contact and **BOUNDARY* in two dimensions.
overcon_bc_contact_3d.inp	Normal contact and **BOUNDARY* in three dimensions.
overcon_bc_friction_2d.inp	Lagrange frictional contact and **BOUNDARY* in two dimensions.
overcon_bc_friction_3d.inp	Lagrange frictional contact and **BOUNDARY* in three dimensions.

Thermal expansion of constraints

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Problem description

A circular plate with a circular hole at the center that is free to expand is modeled. Various types of constraints are used that couple the inside surface of the hole to the reference node at the center of the plate. A uniform increase in temperature is prescribed at all nodes including the reference node of the constraint.

Results and discussion

A uniform expansion of the plate and uniform expansion of the constraint is observed. No stresses are generated in the plate.

Input files

ver_thermalexpkinematic.inp	Three-dimensional solid element model with kinematic coupling that uses the perferred interface to define the kinematic coupling.
ver_thermalexpkinematiccoup.inp	Three-dimensional solid element model with kinematic coupling that uses the nonpreferred interface to define the kinematic coupling.
ver_thermalexpkinematicshell.inp	Three-dimensional shell element model with kinematic coupling.
ver_thermalexpmpc.inp	Three-dimensional solid element model with a MPC BEAM.
ver_thermalexpmpclink.inp	Three-dimensional solid element model with a MPC LINK.
ver_thermalrigid.inp	Three-dimensional solid element model with a rigid body.
ver_thermalexpkine2d.inp	Two-dimensional solid element model with kinematic coupling.

Elements Verification

In this section:

- *About element verification tests*
- *Eigenvalue tests*
- *Simple load tests*
- *Element loading options*
- *Patch tests*
- *Connector element verification*
- *Special-purpose stress/displacement elements*
- *Miscellaneous tests*

About element verification tests

Overview

This section defines the basic tests used to provide evidence that the implementation of the numerical model for the behavior of the elements in Abaqus produces the expected results. Verification of various print and file output options is also provided in these tests.

The test set is divided into categories as described below.

Eigenvalue tests

This set includes two tests for most element types. In the first of these tests all the modes and frequencies of a single, unrestrained element are extracted. The second test extracts the modes and frequencies of a patch of unrestrained elements. These tests verify the correct representation of rigid body modes and the correctness of each element's stiffness and mass. The tests also reveal any singular “hourglass” modes that may be present in reduced-integration elements.

A third test is performed to extract the natural modes of vibration of an organ pipe modeled with acoustic elements.

Only the number of zero-energy modes has been verified for the tests. The first nonzero eigenvalue is shown only for purposes of comparison. These tests are not performed for heat transfer elements and some other nonstructural elements.

Simple load tests

In these tests a simple domain, such as a rectangle in two dimensions or a rectangular prism in three dimensions, is discretized with the minimum number of elements. Sufficient kinematic boundary conditions are imposed to remove rigid body motion only. The loadings that are applied are ones for which the element being tested is capable of representing the solution exactly; for example, first-order elements are loaded so as to cause a constant stress state, while second-order elements are loaded into a linearly varying stress state. The results are checked against exact calculations.

Several such tests are necessary for structural elements (beams and shells) because of their complexity, and different tests are used for the elements that are based on the Kirchhoff hypothesis and for those that provide shear flexibility. The tests also include discontinuous structures (plates joined at an angle and frames) to test defining discontinuous normals, and they include shells and membranes with variable thickness. The local coordinate definitions are verified in some tests.

The problem descriptions contain the solution with which the results are compared. Where analytical solutions are not available, alternative numerical solutions are used.

Element loading options

In these tests the distributed loadings provided for each element are verified by checking the equivalent nodal forces, fluxes, or charges that are calculated for each load type. All degrees of freedom are suppressed, and the various distributed loadings offered for the element type are applied in a series of steps. The reactions are verified against exact calculation for the interpolation function. The values of the output variables presented are “exact” in the finite element sense and, unless noted otherwise, are also exact in the analytical sense.

To check thermal loading, free and constrained thermal expansions of elements are also tested. Thermal loads are defined by giving the temperature, θ, along with a nonzero thermal expansion coefficient.

Generalized plane strain elements have an additional reference node associated with the generalized plane strain condition. Depending on the particular test, degrees of freedom u_z, ϕ_x, and ϕ_y of the generalized plane strain reference node are constrained or left free.

Patch tests

The patch test requires that, for an arbitrary "patch" of elements, when a solution corresponding to a state of constant strain throughout the patch is prescribed on the boundary of the patch, the constant strain state must be obtained as the solution at all strain calculation points throughout the patch. For heat transfer elements the patch test requires that constant temperature gradients are calculated throughout the patch when the temperatures corresponding to the constant gradient solution are prescribed on the boundary. The acoustic elements are similarly tested for constant pressure gradients, and the thermal-electrical elements are tested for constant potential gradients.

The patch test is generally considered to be a necessary and sufficient condition for convergence of the solution as the element size is reduced, except for shell elements of the type used in Abaqus, for which the test is not rigorously required, but for which it is commonly accepted as a valuable indicator of the element's quality. Thus, this test plays a key role in the verification process.

In the patch tests done in Abaqus a patch is defined as a mesh with at least one interior element and several interior nodes. The elements in the patch are nonrectangular, although element edges are kept straight. (Second-order elements do not always pass the patch test if their edges are not straight.) The shell elements are tested for plate and cylindrical patches only.

Basic verification of the geometric nonlinearity capability is included in these tests by prescribing large rigid body rotations of the models under states of constant strain and verifying the invariance of the solution with respect to the rotation.

Contact tests

This section contains tests of the various contact capabilities available in Abaqus.

Interface tests

This section contains tests of the various interface capabilities available in Abaqus. This category currently consists of modeling surface interface conditions in heat transfer problems, coupled acoustic-structural problems, coupled thermal-electrical problems, and friction.

Rigid body verification

This section contains tests of the rigid body elements available in Abaqus/Explicit.

Connector element verification

This section contains tests of the connector elements available in Abaqus.

Special-purpose stress/displacement elements

This section describes tests of some of the special-purpose stress/displacement elements available in Abaqus that are not tested in other sections of this guide. SPRING- and MASS-type elements are tested with the eigenvalue frequency analyses of *Eigenvalue extraction for single unconstrained elements*. ELBOW-type elements are also

tested in *Eigenvalue extraction for single unconstrained elements*, as well as in the simple load test described in *Verification of beam elements and section types* and the distributed load test described in *ELBOW elements*. GAP-type elements are tested with the contact elements, as described in *Contact between discrete points*.

Miscellaneous tests

This category contains tests of the rebar options, transport of a temperature pulse in convection elements, transverse shear for shear-flexible shells, and linear dynamic analyses with fluid link elements.

Eigenvalue tests

In this section:

- *Eigenvalue extraction for single unconstrained elements*
- *Eigenvalue extraction for unconstrained patches of elements*
- *Acoustic modes*

Eigenvalue extraction for single unconstrained elements

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

Acoustic elements, beams, cohesive elements, elbows, membranes, pipes, shells, trusses, continuum elements (except coupled pore pressure-displacement and coupled temperature-displacement elements), piezoelectric elements, springs, and masses.

Problem description

The models consist of a single element. There are no boundary conditions, except as required in spring-mass (see *SPRING, MASS, and JOINT2D elements*) and piezoelectric tests. For the piezoelectric element tests one electric potential degree of freedom is constrained to remove singularities from the dielectric portion of the structural stiffness.



Note:

There are no mass terms associated with potential degrees of freedom.

Results and discussion

The results presented in *Table 1* through *Table 7* show the number of zero-energy modes and the first nonzero eigenvalue. Some elements have nonrigid-body zero-energy modes. Where two values are given in the zero-energy modes column, the first is the number of zero-energy modes and the second is the number of rigid-body zero-energy modes. When an assembly of elements is tested, as in *Eigenvalue extraction for unconstrained patches of elements*, the nonrigid-body zero-energy modes disappear. The eigenvalue is shown only for purposes of comparison. Elements with quadrilateral geometry can be degenerated to triangular shape; these results are denoted by "(triangle)" in the tables. Results for the piezoelectric elements are reported for Step 2.

Table 1: Acoustic elements.

Element type	Number of zero-energy modes	First nonzero eigenvalue
AC1D2	1	1.509×10^8
AC1D3	1	4.527×10^8
AC2D3	1	1.122×10^8
AC2D4 (triangle)	1	1.122×10^8
AC2D4	1	9.971×10^7
AC2D6	1	4.116×10^8
AC2D8 (triangle)	1	4.077×10^8
AC2D8	1	4.447×10^8

Element type	Number of zero-energy modes	First nonzero eigenvalue
AC3D4	1	1.482×10^{8}
AC3D5	1	9.94×10^{7}
AC3D6	1	4.447×10^{8}
AC3D8	1	3.743×10^{7}
AC3D10	1	5.775×10^{8}
AC3D15	1	4.447×10^{8}
AC3D20	1	1.132×10^{8}
ACAX3	1	1.218×10^{8}
ACAX4 (triangle)	1	1.218×10^{8}
ACAX4	1	9.331×10^{7}
ACAX6	1	4.887×10^{8}
ACAX8 (triangle)	1	4.870×10^{8}
ACAX8	1	4.527×10^{8}

Table 2: Beam elements.

Element type	Number of zero-energy modes	First nonzero eigenvalue
B21	3	1.675×10^{9}
B21H	3	1.675×10^{9}
B22	3	4.621×10^{9}
B22H	3	4.621×10^{9}
B23	3	1.379×10^{10}
B23H	3	1.379×10^{10}
B31	6	3.127×10^{9}
B31H	6	3.127×10^{9}
B31OS	6	8.534×10^{7}
B31OSH	6	8.534×10^{7}
B32	6	7.170×10^{9}
B32H	6	7.170×10^{9}
B32OS	6	2.050×10^{8}
B32OSH	6	2.050×10^{8}
B33	6	1.714×10^{10}
B33H	6	1.714×10^{10}

Table 3: Cohesive elements.

Element type	Number of zero-Energy modes	First nonzero eigenvalue
COH2D4	5/3	1.0256×10^{6}
COHAX4	5/1	1.0256×10^{6}
COH3D6	12/6	1.2820×10^{5}
COH3D8	16/6	5.1282×10^{5}

Table 4: Elbow and pipe elements.

Element type	Number of zero-energy modes	First nonzero eigenvalue
ELBOW31	6	5.481×10^{7}
ELBOW31B	6	3.230×10^{5}
ELBOW31C	6	3.230×10^{5}
ELBOW32	6	1.065×10^{8}
PIPE21	3	1.675×10^{9}
PIPE21H	3	1.675×10^{9}
PIPE22	3	4.621×10^{9}
PIPE22H	3	4.621×10^{9}
PIPE31	6	3.127×10^{9}
PIPE31H	6	3.127×10^{9}
PIPE32	6	9.321×10^{9}
PIPE32H	6	9.321×10^{9}

The membrane elements have no bending stiffness, which accounts for the high number of nonrigid-body zero-energy modes.

Table 5: Membrane elements.

Element type	Number of zero-energy modes	First nonzero eigenvalue
M3D3	6	2.350×10^{8}
M3D4	7	1.615×10^{8}
M3D4R	7	3.140×10^{5}
M3D6	9	3.622×10^{8}
M3D8	11	7.274×10^{8}
M3D8R	12	7.274×10^{8}
M3D9	12	7.274×10^{8}
M3D9R	13	5.225×10^{8}
MAX1	2	1.231×10^{9}
MAX2	2	1.535×10^{9}

Element type	Number of zero-energy modes	First nonzero eigenvalue
MCL6	9	7.582×10^{9}
MCL9	9	6.313×10^{8}

Table 6: Shell elements.

Element type	Number of zero-energy modes	First nonzero eigenvalue
S3/S3R	6	1.985×10^{6}
S4	6	3.071×10^{6}
S4R	6	3.071×10^{6}
S4R5	6	3.074×10^{6}
S8R	8/6	3.073×10^{5}
S8R5	7/6	1.165×10^{4}
S9R5	7/6	1.165×10^{4}
STRI3	6	7.189×10^{7}
STRI65	6	3.049×10^{5}
SAXA11	4/3	1.228×10^{5}
SAXA12	5/3	1.229×10^{5}
SAXA13	6/3	1.229×10^{5}
SAXA14	7/3	1.229×10^{5}
SAXA21	3	2.636×10^{6}
SAXA22	3	4.075×10^{5}
SAXA23	3	4.075×10^{5}
SAXA24	3	4.075×10^{5}
SAX1	2/1	1.231×10^{9}
SAX2	1	2.636×10^{6}
SC6R	6	1.942×10^{8}
SC8R	6	1.942×10^{8}

Table 7: Truss elements.

Element type	Number of zero-energy modes	First nonzero eigenvalue
T2D2	3	1.143×10^{10}
T2D2H	3	1.143×10^{10}
T2D3	4/3	3.429×10^{10}
T2D3H	4/3	3.429×10^{10}
T3D2	5	1.143×10^{10}
T3D2H	5	1.143×10^{10}

Element type	Number of zero-energy modes	First nonzero eigenvalue
T3D3	7/6	3.429×10^{10}
T3D3H	7/6	3.429×10^{10}

Table 8: Two-dimensional continuum elements.

Element type	Number of zero-energy modes	First nonzero eigenvalue
CPE3	3	2.488×10^{8}
CPE3H	3	2.488×10^{8}
CPE4	3	8.373×10^{7}
CPE4H	3	8.373×10^{7}
CPE4I	3	1.196×10^{8}
CPE4IH	3	1.196×10^{8}
CPE4R	3	3.140×10^{5}
CPE4RH	3	3.140×10^{5}
CPE6	3	3.868×10^{8}
CPE6H	3	3.868×10^{8}
CPE6M	3	1.289×10^{8}
CPE6MH	3	1.289×10^{8}
CPE8	3	7.535×10^{8}
CPE8H	3	5.024×10^{8}
CPE8R	4/3	7.535×10^{8}
CPE8RH	4/3	7.535×10^{8}
CPEG3	5/3	4.662×10^{8}
CPEG3H	5/3	4.662×10^{8}
CPEG4	5/3	8.373×10^{7}
CPEG4H	5/3	8.373×10^{7}
CPEG4I	3	1.086×10^{8}
CPEG4IH	3	1.086×10^{8}
CPEG4R	5/3	3.140×10^{5}
CPEG4RH	5/3	3.140×10^{5}
CPEG6	3	3.599×10^{8}
CPEG6H	3	3.599×10^{8}
CPEG8	3	7.168×10^{8}
CPEG8H	3	5.024×10^{8}
CPEG8R	4/3	7.168×10^{8}
CPEG8RH	4/3	7.168×10^{8}
CPS3	3	2.350×10^{8}
CPS4	3	1.615×10^{8}

Element type	Number of zero-energy modes	First nonzero eigenvalue
CPS4I	3	1.088×10^{8}
CPS4R	3	3.140×10^{5}
CPS6	3	3.622×10^{8}
CPS6M	3	1.206×10^{8}
CPS8	3	7.274×10^{8}
CPS8R	4/3	7.274×10^{8}

Table 9: Axisymmetric continuum elements.

Element type	Number of zero-energy modes	First nonzero eigenvalue
CAXA41	4/3	2.015×10^{8}
CAXA42	4/3	4.887×10^{7}
CAXA43	4/3	4.887×10^{7}
CAXA44	4/3	4.887×10^{7}
CAXA4H1	4/3	2.015×10^{8}
CAXA4H2	4/3	4.887×10^{7}
CAXA4H3	4/3	4.887×10^{7}
CAXA4H4	4/3	4.887×10^{7}
CAXA4R1	5/3	9.615×10^{6}
CAXA4R2	8/3	9.615×10^{6}
CAXA4R3	11/3	9.615×10^{6}
CAXA4R4	14/3	9.615×10^{6}
CAXA4RH1	5/3	9.615×10^{6}
CAXA4RH2	8/3	9.615×10^{6}
CAXA4RH3	11/3	9.615×10^{6}
CAXA4RH4	14/3	9.615×10^{6}
CAXA81	3	2.437×10^{8}
CAXA82	3	8.526×10^{7}
CAXA83	3	8.526×10^{7}
CAXA84	3	8.526×10^{7}
CAXA8H1	3	2.156×10^{8}
CAXA8H2	3	8.461×10^{7}
CAXA8H3	3	8.461×10^{7}
CAXA8H4	3	8.461×10^{7}
CAXA8R1	5/3	2.405×10^{8}
CAXA8R2	6/3	8.457×10^{7}
CAXA8R3	7/3	8.457×10^{7}
CAXA8R4	8/3	8.457×10^{7}

Element type	Number of zero-energy modes	First nonzero eigenvalue
CAXA8RH1	5/3	2.099×10^{8}
CAXA8RH2	6/3	8.384×10^{7}
CAXA8RH3	7/3	8.384×10^{7}
CAXA8RH4	8/3	8.348×10^{7}
CAX3	2/1	7.402×10^{8}
CAX3H	2/1	7.402×10^{8}
CAX4	2/1	1.022×10^{9}
CAX4H	2/1	1.022×10^{9}
CAX4R	2/1	1.011×10^{7}
CAX4RH	2/1	1.011×10^{7}
CAX4I	1	7.711×10^{7}
CAX4IH	1	7.456×10^{7}
CAX6	1	1.448×10^{8}
CAX6H	1	1.448×10^{8}
CAX6M	1	8.949×10^{7}
CAX6MH	1	8.949×10^{7}
CAX8	1	2.437×10^{8}
CAX8H	1	2.156×10^{8}
CAX8R	2/1	2.405×10^{8}
CAX8RH	2/1	2.099×10^{8}

Table 10: Three-dimensional continuum elements.

Element type	Number of zero-energy modes	First nonzero eigenvalue
C3D10	6	4.500×10^{9}
C3D10H	6	2.346×10^{4}
C3D10HS	6	4.500×10^{9}
C3D10M	6	6.196×10^{7}
C3D10MH	6	6.196×10^{7}
C3D15	6	1.695×10^{9}
C3D15H	6	1.967×10^{9}
C3D15V	6	1.084×10^{9}
C3D15VH	6	1.379×10^{8}
C3D20	6	3.436×10^{8}
C3D20H	6	2.213×10^{8}
C3D20R	12/6	3.768×10^{8}
C3D20RH	12/6	4.082×10^{3}
C3D27 (21 nodes)	6	3.768×10^{8}

Element type	Number of zero-energy modes	First nonzero eigenvalue
C3D27 (22 nodes)	6	3.768×10^{8}
C3D27 (23 nodes)	6	3.768×10^{8}
C3D27 (24 nodes)	6	3.768×10^{8}
C3D27 (25 nodes)	6	3.768×10^{8}
C3D27 (26 nodes)	6	3.768×10^{8}
C3D27 (27 nodes)	6	3.768×10^{8}
C3D27H (21 nodes)	6	2.213×10^{8}
C3D27H (22 nodes)	6	2.213×10^{8}
C3D27H (23 nodes)	6	2.213×10^{8}
C3D27H (24 nodes)	6	2.213×10^{8}
C3D27H (25 nodes)	6	2.213×10^{8}
C3D27H (26 nodes)	6	2.213×10^{8}
C3D27H (27 nodes)	6	2.213×10^{8}
C3D27R (21 nodes)	6	3.768×10^{8}
C3D27R (22 nodes)	6	3.768×10^{8}
C3D27R (23 nodes)	6	3.768×10^{8}
C3D27R (24 nodes)	6	3.128×10^{8}
C3D27R (25 nodes)	6	1.558×10^{8}
C3D27R (26 nodes)	6	1.236×10^{8}
C3D27R (27 nodes)	9/6	2.007×10^{8}
C3D27RH (21 nodes)	6	2.213×10^{8}
C3D27RH (22 nodes)	6	2.032×10^{8}
C3D27RH (23 nodes)	6	1.467×10^{8}
C3D27RH (24 nodes)	6	1.022×10^{8}
C3D27RH (25 nodes)	6	2.767×10^{7}
C3D27RH (26 nodes)	6	2.509×10^{7}
C3D27RH (27 nodes)	9/6	3.069×10^{7}
C3D4	6	3.623×10^{9}
C3D4H	6	3.623×10^{9}
C3D5	6	2.026×10^{12}
C3D5H	6	1.936×10^{12}
C3D6	7/6	3.846×10^{8}
C3D6H	7/6	3.472×10^{8}
C3D8	6	4.186×10^{7}
C3D8H	6	4.186×10^{7}
C3D8I	6	4.186×10^{7}
C3D8IH	6	4.186×10^{7}
C3D8R	6	1.184×10^{6}

Element type	Number of zero-energy modes	First nonzero eigenvalue
C3D8RH	6	1.184×10^{6}
CCL9	9/6	1.410×10^{5}
CCL9H	9/6	1.0572
CCL12	6	3.1502×10^{8}
CCL12H	6	3.1502×10^{8}
CCL18	6	1.089×10^{10}
CCL18H	6	4.449×10^{8}
CCL24	6	3.767×10^{9}
CCL24R	9/6	3.394×10^{9}
CCL24H	6	2.213×10^{9}
CCL24RH	9/6	1.214×10^{9}
CSS8	6	4.186×10^{7}

Table 11: Piezoelectric elements.

Element type	Number of zero-energy modes	First nonzero eigenvalue
C3D10E	6	4.825×10^{9}
C3D15E	6	1.695×10^{9}
C3D20E	6	3.768×10^{8}
C3D20RE	12/6	3.768×10^{8}
C3D4E	6	6.092×10^{9}
C3D6E	7/6	3.846×10^{8}
C3D8E	6	4.186×10^{7}
CAX3E	2/1	8.828×10^{8}
CAX4E	2/1	1.169×10^{9}
CAX6E	1	1.604×10^{8}
CAX8E	1	2.556×10^{8}
CAX8RE	2/1	2.522×10^{8}
CPE3E	3	6.567×10^{8}
CPE4E	3	8.373×10^{7}
CPE6E	3	6.006×10^{8}
CPE8E	3	8.246×10^{8}
CPE8RE	4/3	8.246×10^{8}
CPS3E	3	5.024×10^{8}
CPS4E	3	1.615×10^{8}
CPS6E	3	5.265×10^{8}
CPS8E	3	7.797×10^{8}
CPS8RE	4/3	7.797×10^{8}

Element type	Number of zero-energy modes	First nonzero eigenvalue
T2D2E	3	1.476×10^{13}
T2D3E	4/3	1.714×10^{11}
T3D2E	5	1.476×10^{13}
T3D3E	7/6	1.714×10^{11}

SPRING, MASS, and JOINT2D elements

The models for the eigenvalue extraction tests for SPRING and MASS element types are slightly more complex than the tests for the other elements.

Elements of type SPRINGA and MASS are tested together in files *exspame1.inp* and *exspame2.inp*. Three nodes lie along a straight line. One of the nodes is constrained, and each of the other two nodes defines a point mass. SPRINGA elements are defined between each of the three possible pairs of nodes. The spring-mass system acts in degree of freedom 1.

File *exspbue1.inp* tests element types SPRING1 and SPRING2 with a mass matrix defined by a user element. Two coincident nodes are defined. These two nodes are used in the definition of the user element. A SPRING2 element connects the nodes, and each node is also connected to a SPRING1 element. No boundary conditions are required since, by definition, the other ends of the SPRING1 elements are connected to ground. The spring-mass system acts in degree of freedom 1.

Results for both tests: ω_1=0.6340, ω_2=2.3660.

File *exepxme1.inp* tests element type JOINT2D. One node of the JOINT2D element is fully constrained, and the other has MASS and ROTARYI elements applied to create a spring-mass system. The natural frequencies and modes correspond to analytically calculated values.

Input files

Acoustic elements

ec12afe1.inp	AC1D2 elements.
ec13afe1.inp	AC1D3 elements.
ec23afe1.inp	AC2D3 elements.
ec24afe1t.inp	AC2D4 elements (triangle).
ec24afe1.inp	AC2D4 elements.
ec26afe1.inp	AC2D6 elements.
ec28afe1t.inp	AC2D8 elements (triangle).
ec28afe1.inp	AC2D8 elements.
ec34afe1.inp	AC3D4 elements.
ec35afe1.inp	AC3D5 elements.
ec36afe1.inp	AC3D6 elements.
ec38afe1.inp	AC3D8 elements.
ec3aafe1.inp	AC3D10 elements.
ec3fafe1.inp	AC3D15 elements.
ec3kafe1.inp	AC3D20 elements.
ec34afe1_ams.inp	AC3D4 elements, Abaqus/AMS.

ec36afe1_ams.inp	AC3D6 elements, Abaqus/AMS.
ec38afe1_ams.inp	AC3D8 elements, Abaqus/AMS.
ec3aafe1_ams.inp	AC3D10 elements, Abaqus/AMS.
ec3fafe1_ams.inp	AC3D15 elements, Abaqus/AMS.
ec3kafe1.inp	AC3D20 elements, Abaqus/AMS.
eca3afe1.inp	ACAX3 elements.
eca4afe1t.inp	ACAX4 elements (triangle).
eca4afe1.inp	ACAX4 elements.
eca6afe1.inp	ACAX6 elements.
eca8afe1t.inp	ACAX8 elements (triangle).
eca8afe1.inp	ACAX8 elements.
eca3afe1_ams.inp	ACAX3 elements, Abaqus/AMS.
eca4afe1t_ams.inp	ACAX4 elements (triangle).
eca4afe1_ams.inp	ACAX4 elements, Abaqus/AMS.
eca6afe1_ams.inp	ACAX6 elements, Abaqus/AMS.
eca8afe1t_ams.inp	ACAX8 elements (triangle), Abaqus/AMS.
eca8afe1_ams.inp	ACAX8 elements, Abaqus/AMS.

Beam elements

eb22pxe1.inp	B21 elements.
eb2hpxe1.inp	B21H elements.
eb23pxe1.inp	B22 elements.
eb2ipxe1.inp	B22H elements.
eb2apxe1.inp	B23 elements.
eb2jpxe1.inp	B23H elements.
eb32pxe1.inp	B31 elements.
eb3hpxe1.inp	B31H elements.
ebo2ixe1.inp	B31OS elements.
ebohixe1.inp	B31OSH elements.
eb33pxe1.inp	B32 elements.
eb3ipxe1.inp	B32H elements.
ebo3ixe1.inp	B32OS elements.
eboiixe1.inp	B32OSH elements.
eb3apxe1.inp	B33 elements.
eb3jpxe1.inp	B33H elements.

Cohesive elements

coh2d4_eig.inp	COH2D4 elements.
cohax4_eig.inp	COHAX4 elements.
coh3d6_eig.inp	COH3D6 elements.
coh3d8_eig.inp	COH3D8 elements.

Elbow and pipe elements

exel1xe1.inp	ELBOW31 elements.
exelbxe1.inp	ELBOW31B elements.
exelcxe1.inp	ELBOW31C elements.
exel2xe1.inp	ELBOW32 elements.
ep22pxe1.inp	PIPE21 elements.
ep2hpxe1.inp	PIPE21H elements.
ep23pxe1.inp	PIPE22 elements.
ep2ipxe1.inp	PIPE22H elements.
ep32pxe1.inp	PIPE31 elements.
ep3hpxe1.inp	PIPE31H elements.
ep33pxe1.inp	PIPE32 elements.
ep3ipxe1.inp	PIPE32H elements.

Membrane elements

em33sfe1.inp	M3D3 elements.
em34sfe1.inp	M3D4 elements.
em34sre1.inp	M3D4R elements.
em36sfe1.inp	M3D6 elements.
em38sfe1.inp	M3D8 elements.
em38sre1.inp	M3D8R elements.
em39sfe1.inp	M3D9 elements.
em39sre1.inp	M3D9R elements.
ema2sre1.inp	MAX1 elements.
ema3sre1.inp	MAX2 elements.
emc6sre1.inp	MCL6 elements.
emc9sre1.inp	MCL9 elements.

Shell elements

esf3sxe1.inp	S3/S3R elements.
ese4sxe1.inp	S4 elements.
esf4sxe1.inp	S4R elements.
es54sxe1.inp	S4R5 elements.
es68sxe1.inp	S8R elements.
es58sxe1.inp	S8R5 elements.
es59sxe1.inp	S9R5 elements.
es63sxe1.inp	STRI3 elements.
es56sxe1.inp	STRI65 elements.
esnssxe1.inp	SAXA11 elements.
esntsxe1.inp	SAXA12 elements.
esnusxe1.inp	SAXA13 elements.
esnvsxe1.inp	SAXA14 elements.
esnwsxe1.inp	SAXA21 elements.

esnxsxe1.inp	SAXA22 elements.
esnysxe1.inp	SAXA23 elements.
esnzsxe1.inp	SAXA24 elements.
esa2sxe1.inp	SAX1 elements.
esa3sxe1.inp	SAX2 elements.
esc6sxe1.inp	SC6R elements.
esc8sxe1.inp	SC8R elements.

Truss elements

et22sfe1.inp	T2D2 elements.
et22she1.inp	T2D2H elements.
et23sfe1.inp	T2D3 elements.
et23she1.inp	T2D3H elements.
et32sfe1.inp	T3D2 elements.
et32she1.inp	T3D2H elements.
et33sfe1.inp	T3D3 elements.
et33she1.inp	T3D3H elements.

Two-dimensional continuum elements

ece3sfe1.inp	CPE3 elements.
ece3she1.inp	CPE3H elements.
ece4sfe1.inp	CPE4 elements.
ece4she1.inp	CPE4H elements.
ece4sie1.inp	CPE4I elements.
ece4sje1.inp	CPE4IH elements.
ece4sre1.inp	CPE4R elements.
ece4sye1.inp	CPE4RH elements.
ece6sfe1.inp	CPE6 elements.
ece6she1.inp	CPE6H elements.
ece6ske1.inp	CPE6M elements.
ece6sle1.inp	CPE6MH elements.
ece8sfe1.inp	CPE8 elements.
ece8she1.inp	CPE8H elements.
ece8sre1.inp	CPE8R elements.
ece8sye1.inp	CPE8RH elements.
ecg3sfe1.inp	CPEG3 elements.
ecg3she1.inp	CPEG3H elements.
ecg4sfe1.inp	CPEG4 elements.
ecg4she1.inp	CPEG4H elements.
ecg4sie1.inp	CPEG4I elements.
ecg4sje1.inp	CPEG4IH elements.
ecg4sre1.inp	CPEG4R elements.
ecg4sye1.inp	CPEG4RH elements.

ecg6sfe1.inp	CPEG6 elements.
ecg6she1.inp	CPEG6H elements.
ecg8sfe1.inp	CPEG8 elements.
ecg8she1.inp	CPEG8H elements.
ecg8sre1.inp	CPEG8R elements.
ecg8sye1.inp	CPEG8RH elements.
ecs3sfe1.inp	CPS3 elements.
ecs4sfe1.inp	CPS4 elements.
ecs4sie1.inp	CPS4I elements.
ecs4sre1.inp	CPS4R elements.
ecs6sfe1.inp	CPS6 elements.
ecs6ske1.inp	CPS6M elements.
ecs8sfe1.inp	CPS8 elements.
ecs8sre1.inp	CPS8R elements.

Axisymmetric continuum elements

ecnssfe1.inp	CAXA41 elements.
ecntsfe1.inp	CAXA42 elements.
ecnusfe1.inp	CAXA43 elements.
ecnvsfe1.inp	CAXA44 elements.
ecnsshe1.inp	CAXA4H1 elements.
ecntshe1.inp	CAXA4H2 elements.
ecnushe1.inp	CAXA4H3 elements.
ecnvshe1.inp	CAXA4H4 elements.
ecnssre1.inp	CAXA4R1 elements.
ecntsre1.inp	CAXA4R2 elements.
ecnusre1.inp	CAXA4R3 elements.
ecnvsre1.inp	CAXA4R4 elements.
ecnssye1.inp	CAXA4RH1 elements.
ecntsye1.inp	CAXA4RH2 elements.
ecnusye1.inp	CAXA4RH3 elements.
ecnvsye1.inp	CAXA4RH4 elements.
ecnwsfe1.inp	CAXA81 elements.
ecnxsfe1.inp	CAXA82 elements.
ecnysfe1.inp	CAXA83 elements.
ecnzsfe1.inp	CAXA84 elements.
ecnwshe1.inp	CAXA8H1 elements.
ecnxshe1.inp	CAXA8H2 elements.
ecnyshe1.inp	CAXA8H3 elements.
ecnzshe1.inp	CAXA8H4 elements.
ecnwsre1.inp	CAXA8R1 elements.
ecnxsre1.inp	CAXA8R2 elements.
ecnysre1.inp	CAXA8R3 elements.

ecnzsre1.inp	CAXA8R4 elements.
ecnwsye1.inp	CAXA8RH1 elements.
ecnxsye1.inp	CAXA8RH2 elements.
ecnysye1.inp	CAXA8RH3 elements.
ecnzsye1.inp	CAXA8RH4 elements.
eca3sfe1.inp	CAX3 elements.
eca3she1.inp	CAX3H elements.
eca4sfe1.inp	CAX4 elements.
eca4she1.inp	CAX4H elements.
eca4sie1.inp	CAX4I elements.
eca4sje1.inp	CAX4IH elements.
eca4sre1.inp	CAX4R elements.
eca4sye1.inp	CAX4RH elements.
eca6sfe1.inp	CAX6 elements.
eca6she1.inp	CAX6H elements.
eca6ske1.inp	CAX6M elements.
eca6sle1.inp	CAX6MH elements.
eca8sfe1.inp	CAX8 elements.
eca8she1.inp	CAX8H elements.
eca8sre1.inp	CAX8R elements.
eca8sye1.inp	CAX8RH elements.

Three-dimensional continuum elements

ec3asfe1.inp	C3D10 elements.
ec3ashe1.inp	C3D10H elements.
ec3asie1.inp	C3D10HS elements.
ec3aske1.inp	C3D10M elements.
ec3asle1.inp	C3D10MH elements.
ec3fsfe1.inp	C3D15 elements.
ec3fshe1.inp	C3D15H elements.
ec3isfe1.inp	C3D15V elements.
ec3ishe1.inp	C3D15VH elements.
ec3ksfe1.inp	C3D20 elements.
ec3kshe1.inp	C3D20H elements.
ec3ksre1.inp	C3D20R elements.
ec3ksye1.inp	C3D20RH elements.
ec3rsfea.inp	C3D27 elements, 21 nodes.
ec3rsfeb.inp	C3D27 elements, 22 nodes.
ec3rsfec.inp	C3D27 elements, 23 nodes.
ec3rsfed.inp	C3D27 elements, 24 nodes.
ec3rsfee.inp	C3D27 elements, 25 nodes.
ec3rsfef.inp	C3D27 elements, 26 nodes.
ec3rsfeg.inp	C3D27 elements, 27 nodes.

ec3rshea.inp	C3D27H elements, 21 nodes.
ec3rsheb.inp	C3D27H elements, 22 nodes.
ec3rshec.inp	C3D27H elements, 23 nodes.
ec3rshed.inp	C3D27H elements, 24 nodes.
ec3rshee.inp	C3D27H elements, 25 nodes.
ec3rshef.inp	C3D27H elements, 26 nodes.
ec3rsheg.inp	C3D27H elements, 27 nodes.
ec3rsrea.inp	C3D27R elements, 21 nodes.
ec3rsreb.inp	C3D27R elements, 22 nodes.
ec3rsrec.inp	C3D27R elements, 23 nodes.
ec3rsred.inp	C3D27R elements, 24 nodes.
ec3rsree.inp	C3D27R elements, 25 nodes.
ec3rsref.inp	C3D27R elements, 26 nodes.
ec3rsreg.inp	C3D27R elements, 27 nodes.
ec3rsyea.inp	C3D27RH elements, 21 nodes.
ec3rsyeb.inp	C3D27RH elements, 22 nodes.
ec3rsyec.inp	C3D27RH elements, 23 nodes.
ec3rsyed.inp	C3D27RH elements, 24 nodes.
ec3rsyee.inp	C3D27RH elements, 25 nodes.
ec3rsyef.inp	C3D27RH elements, 26 nodes.
ec3rsyeg.inp	C3D27RH elements, 27 nodes.
ec34sfe1.inp	C3D4 elements.
ec34she1.inp	C3D4H elements.
ec35sfe1.inp	C3D5 elements.
ec35she1.inp	C3D5H elements.
ec36sfe1.inp	C3D6 elements.
ec36she1.inp	C3D6H elements.
ec38sfe1.inp	C3D8 elements.
ec38she1.inp	C3D8H elements.
ec38sie1.inp	C3D8I elements.
ec38sje1.inp	C3D8IH elements.
ec38sre1.inp	C3D8R elements.
ec38sye1.inp	C3D8RH elements.
ecc9gfe1.inp	CCL9 elements.
ecc9ghe1.inp	CCL9H elements.
ecccgfe1.inp	CCL12 elements.
ecccghe1.inp	CCL12H elements.
eccigfe1.inp	CCL18 elements.
eccighe1.inp	CCL18H elements.
eccrgfe1.inp	CCL24 elements.
eccrgre1.inp	CCL24R elements.
eccrghe1.inp	CCL24H elements.
eccrgye1.inp	CCL24RH elements.

ecss8sie1.inp	CSS8 elements.

Piezoelectric elements

ec3aefe1.inp	C3D10E elements.
ec3fefe1.inp	C3D15E elements.
ec3kefe1.inp	C3D20E elements.
ec3kere1.inp	C3D20RE elements.
ec34efe1.inp	C3D4E elements.
ec36efe1.inp	C3D6E elements.
ec38efe1.inp	C3D8E elements.
eca3efe1.inp	CAX3E elements.
eca4efe1.inp	CAX4E elements.
eca6efe1.inp	CAX6E elements.
eca8efe1.inp	CAX8E elements.
eca8ere1.inp	CAX8RE elements.
ece3efe1.inp	CPE3E elements.
ece4efe1.inp	CPE4E elements.
ece6efe1.inp	CPE6E elements.
ece8efe1.inp	CPE8E elements.
ece8ere1.inp	CPE8RE elements.
ecs3efe1.inp	CPS3E elements.
ecs4efe1.inp	CPS4E elements.
ecs6efe1.inp	CPS6E elements.
ecs8efe1.inp	CPS8E elements.
ecs8ere1.inp	CPS8RE elements.
et22efe1.inp	T2D2E elements.
et23efe1.inp	T2D3E elements.
et32efe1.inp	T3D2E elements.
et33efe1.inp	T3D3E elements.

Spring, mass, and joint elements

exepxme1.inp	JOINT2D elements.
exspame1.inp	SPRINGA and MASS elements.
exspame2.inp	SPRINGA and MASS elements with anisotropic mass.
exspbue1.inp	SPRING1 and SPRING2 elements.

Eigenvalue extraction for unconstrained patches of elements

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Continuum elements

Elements tested

Continuum elements (excluding coupled temperature-displacement and pore pressure elements).

Problem description

The models consist of the same patches of elements used in the tests defined in *Patch tests*. The first step consists of an eigenvalue analysis of the model with no boundary conditions. The second step applies a uniform pressure load on all four edges and sets the NLGEOM parameter. The third step performs an eigenvalue analysis of the prestressed model with no boundary conditions. Results are printed only for the first and third steps.

Results and discussion

For all elements the number of zero-energy modes for Steps 1 and 3 is the same and matches the number of rigid-body modes given in *Eigenvalue extraction for single unconstrained elements*.

Input files

ec3asfe2.inp	C3D10 elements.
ec3ashe2.inp	C3D10H elements.
ec3asie2.inp	C3D10HS elements.
ec3aske2.inp	C3D10M elements.
ec3asle2.inp	C3D10MH elements.
ec3fsfe2.inp	C3D15 elements.
ec3fshe2.inp	C3D15H elements.
ec3isfe2.inp	C3D15V elements.
ec3ishe2.inp	C3D15VH elements.
ec3ksfe2.inp	C3D20 elements.
ec3kshe2.inp	C3D20H elements.
ec3ksre2.inp	C3D20R elements.
ec3ksye2.inp	C3D20RH elements.
ec3rsfe2.inp	C3D27 elements.
ec3rshe2.inp	C3D27H elements.
ec3rsre2.inp	C3D27R elements.
ec3rsye2.inp	C3D27RH elements.
ec34sfe2.inp	C3D4 elements.
ec34she2.inp	C3D4H elements.

ec35sfe2.inp	C3D5 elements.
ec35she2.inp	C3D5H elements.
ec36sfe2.inp	C3D6 elements.
ec36she2.inp	C3D6H elements.
ec38sfe2.inp	C3D8 elements.
ec38she2.inp	C3D8H elements.
ec38sie2.inp	C3D8I elements.
ec38sje2.inp	C3D8IH elements.
ec38sre2.inp	C3D8R elements.
ec38sye2.inp	C3D8RH elements.
eca3sfe2.inp	CAX3 elements.
eca3she2.inp	CAX3H elements.
eca4sfe2.inp	CAX4 elements.
eca4she2.inp	CAX4H elements.
eca4sie2.inp	CAX4I elements.
eca4sje2.inp	CAX4IH elements.
eca4sre2.inp	CAX4R elements.
eca4sye2.inp	CAX4RH elements.
eca6sfe2.inp	CAX6 elements.
eca6she2.inp	CAX6H elements.
eca6ske2.inp	CAX6M elements.
eca6sle2.inp	CAX6MH elements.
eca8sfe2.inp	CAX8 elements.
eca8she2.inp	CAX8H elements.
eca8sre2.inp	CAX8R elements.
eca8sye2.inp	CAX8RH elements.
ece3sfe2.inp	CPE3 elements.
ece3she2.inp	CPE3H elements.
ece4sfe2.inp	CPE4 elements.
ece4she2.inp	CPE4H elements.
ece4sie2.inp	CPE4I elements.
ece4sje2.inp	CPE4IH elements.
ece4sre2.inp	CPE4R elements.
ece4sye2.inp	CPE4RH elements.
ece6sfe2.inp	CPE6 elements.
ece6she2.inp	CPE6H elements.
ece6ske2.inp	CPE6M elements.
ece6sle2.inp	CPE6MH elements.
ece8sfe2.inp	CPE8 elements.
ece8she2.inp	CPE8H elements.
ece8sre2.inp	CPE8R elements.
ece8sye2.inp	CPE8RH elements.
ecg3sfe2.inp	CPEG3 elements.

ecg3she2.inp	CPEG3H elements.
ecg4sfe2.inp	CPEG4 elements.
ecg4she2.inp	CPEG4H elements.
ecg4sie2.inp	CPEG4I elements.
ecg4sje2.inp	CPEG4IH elements.
ecg4sre2.inp	CPEG4R elements.
ecg4sye2.inp	CPEG4RH elements.
ecg6sfe2.inp	CPEG6 elements.
ecg6she2.inp	CPEG6H elements.
ecg8sfe2.inp	CPEG8 elements.
ecg8she2.inp	CPEG8H elements.
ecg8sre2.inp	CPEG8R elements.
ecg8sye2.inp	CPEG8RH elements.
ecs3sfe2.inp	CPS3 elements.
ecs4sfe2.inp	CPS4 elements.
ecs4sie2.inp	CPS4I elements.
ecs4sre2.inp	CPS4R elements.
ecs6sfe2.inp	CPS6 elements.
ecs6ske2.inp	CPS6M elements.
ecs8sfe2.inp	CPS8 elements.
ecs8sre2.inp	CPS8R elements.

Beams, pipes, shells

Elements tested

Beams, pipes, general shells.

Problem description

The models consist of the same patches of elements used in the tests defined in *Patch tests*. There are no boundary conditions defined in these models.

Results and discussion

For all elements the number of zero-energy modes matches the number of rigid-body modes given in *Eigenvalue extraction for single unconstrained elements*.

Input files

eb22rxe3.inp	B21 elements.
eb2hrxe3.inp	B21H elements.
eb23rxe3.inp	B22 elements.
eb2irxe3.inp	B22H elements.
eb2arxe3.inp	B23 elements.
eb2jrxe3.inp	B23H elements.
eb32rxe3.inp	B31 elements.
eb3hrxe3.inp	B31H elements.

ebo2ixe3.inp	B31OS elements.
ebohixe3.inp	B31OSH elements.
eb33rxe3.inp	B32 elements.
eb3irxe3.inp	B32H elements.
ebo3ixe3.inp	B32OS elements.
eboiixe3.inp	B32OSH elements.
eb3arxe3.inp	B33 elements.
eb3jrxe3.inp	B33H elements.
ep22pxe3.inp	PIPE21 elements.
ep2hpxe3.inp	PIPE21H elements.
ep23pxe3.inp	PIPE22 elements.
ep2ipxe3.inp	PIPE22H elements.
ep32pxe3.inp	PIPE31 elements.
ep3hpxe3.inp	PIPE31H elements.
ep33pxe3.inp	PIPE32 elements.
ep3ipxe3.inp	PIPE32H elements.
esf3sxe3.inp	S3/S3R elements.
ese4sxe3.inp	S4 elements.
esf4sxe3.inp	S4R elements.
es54sxe3.inp	S4R5 elements.
es68sxe3.inp	S8R elements.
es58sxe3.inp	S8R5 elements.
es59sxe3.inp	S9R5 elements.
es63sxe3.inp	STRI3 elements.
es56sxe3.inp	STRI65 elements.

Acoustic modes

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Organ pipe modes

Elements tested

AC1D2 AC1D3
ACAX3 ACAX4 ACAX6 ACAX8
AC2D3 AC2D4 AC2D6 AC2D8
AC3D4 AC3D5 AC3D6 AC3D8 AC3D10 AC3D15 AC3D20

Features tested

Eigenfrequency extraction

Surface impedance

Problem description

Each member of the family of acoustic elements is used to model an organ pipe. The natural modes of vibration are extracted from the models for the case of an organ pipe with both ends open (open/open) and the case of an organ pipe with one end open and the other end closed (open/closed). The appropriate boundary condition at an open end is that the acoustic pressure degrees of freedom be set to zero (a free surface). A closed end requires no boundary condition; the natural boundary condition is that of a rigid surface adjacent to the fluid. Results are compared with exact solutions.

The model consists of a column of air 165.8 units high with a cross-sectional area of 1.0. The first-order element model consists of 20 acoustic elements along the length of the fluid column and one through the cross-section. The second-order element models consist of 10 elements.

The material properties used for the air are $\rho_f = 1.293$ and bulk modulus = 1.42176×10^5.

Results and discussion

The geometry and material properties defined for this problem result in the natural frequencies of $f_1 = 1.0$ cycles/sec, $f_2 = 2.0$ cycles/sec, and $f_3 = 3.0$ cycles/sec for the open organ pipe and $f_1 = 0.5$ cycles/sec, $f_2 = 1.5$ cycles/sec, and $f_3 = 2.5$ cycles/sec for the closed organ pipe.

The results deviate less than 1% from these frequencies for the first-order elements and less than 0.1% for the second-order elements. More accuracy can be acquired with finer meshes. To match these frequencies with two- and three-dimensional finite elements, the length of the fluid column is chosen considerably longer than the width of the column.

Input files

ec12afe4.inp	AC1D2 elements.
ec13afe4.inp	AC1D3 elements.

eca3afe4.inp	ACAX3 elements.
eca4afe4.inp	ACAX4 elements.
eca6afe4.inp	ACAX6 elements.
eca8afe4.inp	ACAX8 elements.
eca3afe4_ams.inp	ACAX3 elements, Abaqus/AMS.
eca4afe4_ams.inp	ACAX4 elements, Abaqus/AMS.
eca6afe4_ams.inp	ACAX6 elements, Abaqus/AMS.
eca8afe4_ams.inp	ACAX8 elements, Abaqus/AMS.
ec23afe4.inp	AC2D3 elements.
ec24afe4.inp	AC2D4 elements.
ec26afe4.inp	AC2D6 elements.
ec28afe4.inp	AC2D8 elements.
ec34afe4.inp	AC3D4 elements.
ec35afe4.inp	AC3D5 elements.
ec36afe4.inp	AC3D6 elements.
ec38afe4.inp	AC3D8 elements.
ec3aafe4.inp	AC3D10 elements.
ec3fafe4.inp	AC3D15 elements.
ec3kafe4.inp	AC3D20 elements.
ec34afe4_ams.inp	AC3D4 elements, Abaqus/AMS.
ec35afe4_ams.inp	AC3D5 elements, Abaqus/AMS.
ec36afe4_ams.inp	AC3D6 elements, Abaqus/AMS.
ec38afe4_ams.inp	AC3D8 elements, Abaqus/AMS.
ec3aafe4_ams.inp	AC3D10 elements, Abaqus/AMS.
ec3fafe4_ams.inp	AC3D15 elements, Abaqus/AMS.
ec3kafe4_ams.inp	AC3D20 elements, Abaqus/AMS.

Exterior modes with nonreflecting impedance

Elements tested

ACAX3 ACAX4 ACAX6 ACAX8
AC2D3 AC2D4 AC2D6 AC2D8
AC3D4 AC3D5 AC3D6 AC3D8 AC3D10 AC3D15 AC3D20

Problem description

The models consist of duct-like meshes of length 0.1. The first step consists of an eigenvalue analysis of the model with no boundary conditions. The second step applies a spherical nonreflecting impedance on all exterior ends of the ducts. The third step performs an eigenvalue analysis of the model with the impedance conditions. Results are printed only for the first and third steps.

Results and discussion

For all elements the modal analysis results agree with the expected behavior.

Input files

acoustic_exteig2d.inp	AC2D3, AC2D4, AC2D6, and AC2D8 elements.
acoustic_exteigax.inp	ACAX3, ACAX4, ACAX6, and ACAX8 elements.
acoustic_exteig3d.inp	AC3D4, AC3D6, AC3D8, AC3D10, AC3D15, and AC3D20 elements.

Exterior modes with acoustic infinite elements

Elements tested

Acoustic finite elements

ACAX3 ACAX4 ACAX6 ACAX8
AC2D3 AC2D4 AC2D6 AC2D8
AC3D4 AC3D5 AC3D6 AC3D8 AC3D10 AC3D15 AC3D20

Acoustic infinite elements

ACINAX2 ACINAX3
ACIN2D2 ACIN2D3
ACIN3D3 ACIN3D4 ACIN3D6 ACIN3D8

Problem description

The models consist of duct-like meshes of length 0.1, terminated with acoustic infinite elements. The first analysis step consists of a real eigenvalue analysis of the model. The second step performs a complex eigenvalue analysis of the model.

Results and discussion

For all elements the modal analysis results agree with the expected behavior.

Input files

acoustic_infeig2d.inp	ACIN2D2, ACIN2D3, AC2D3, AC2D4, AC2D6, and AC2D8 elements.
acoustic_infeigax.inp	ACINAX2, ACINAX3, ACAX3, ACAX4, ACAX6, and ACAX8 elements.
acoustic_infeig3d.inp	ACIN3D3, ACIN3D4, ACIN3D6, ACIN3D8, AC3D4, AC3D6, AC3D8, AC3D10, AC3D15, and AC3D20 elements.

Simple load tests

In this section:

- *Membrane loading of plane stress, plane strain, membrane, and shell elements*
- *Generalized plane strain elements with relative motion of bounding planes*
- *Three-dimensional solid elements*
- *Axisymmetric solid elements*
- *Axisymmetric solid elements with twist*
- *Cylindrical elements*
- *Loading of piezoelectric elements*
- *Love-Kirchhoff beams and shells*
- *Shear flexible beams and shells: I*
- *Shear flexible beams and shells: II*
- *Initial curvature of beams and shells*
- *Normal definitions of beams and shells*
- *Constant curvature test for shells*
- *Verification of section forces for shells*
- *Composite shell sections*
- *Cantilever sandwich beam: shear flexible shells*
- *Thermal stress in a cylindrical shell*
- *Variable thickness shells and membranes*
- *Shell offset*
- *Axisymmetric membrane elements*
- *Cylindrical membrane elements*
- *Verification of beam elements and section types*
- *Beam added inertia*
- *Beam fluid inertia*
- *Beam with end moment*
- *Flexure of a deep beam*
- *Simple tests of beam kinematics*
- *Tensile test*
- *Simple shear*
- *Verification of the elastic behavior of frame elements*
- *Verification of the plastic behavior of frame elements*
- *Three-bar truss*
- *Pure bending of a cylinder: CAXA elements*
- *Cylinder subjected to an asymmetric temperature field: CAXA elements*
- *Cylinder subjected to asymmetric pressure loads: CAXA elements*
- *Cylinder subjected to an asymmetric pore pressure field: CAXA elements*
- *Modal dynamic and transient dynamic analysis with CAXA and SAXA elements*
- *Simple load tests for thermal-electrical elements*
- *Hydrostatic fluid elements*
- *Fluid link element*

- *Fluid pipe element*
- *Thermal fluid pipe elements*
- *Temperature-dependent film condition*
- *Surface-based pressure penetration*
- *Gasket behavior verification*
- *Gasket element assembly*
- *Cohesive elements*
- *Coriolis loading for direct-solution steady-state dynamic analysis*
- *Pipe-soil interaction elements*
- *Shear panel elements*

Membrane loading of plane stress, plane strain, membrane, and shell elements

Products: Abaqus/Standard

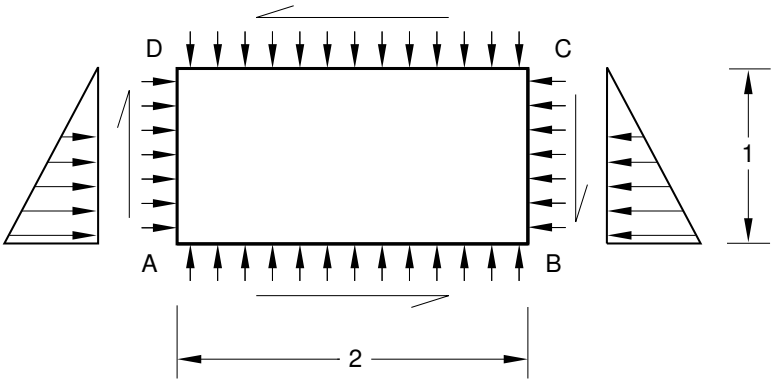
Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

CPS3 CPS4 CPS4I CPS4R CPS4RT CPS6 CPS6M CPS6MT CPS8 CPS8R
CPE3 CPE3H CPE4 CPE4H CPE4I CPE4IH CPE4R CPE4RH CPE4RHT CPE4RT CPE6 CPE6H CPE6M CPE6MH CPE6MHT CPE6MT CPE8 CPE8H CPE8R CPE8RH
CPEG3 CPEG3H CPEG4 CPEG4H CPEG4I CPEG4IH CPEG4R CPEG4RH CPEG6 CPEG6H CPEG6M CPEG6MH CPEG8 CPEG8H CPEG8R CPEG8RH
M3D3 M3D4 M3D4R M3D6 M3D8 M3D8R M3D9 M3D9R
S4 S4R S4R5 S8R S8R5 S9R5 STRI3 STRI65 SC8R CSS8

Problem description



Material: Linear elastic, Young's modulus = 30×10^6, Poisson's ratio = 0.3.

For the coupled temperature-displacement elements dummy thermal properties are prescribed to complete the material definitions.

Boundary conditions: $u_x^A = u_y^A = 0, u_y^B = 0$ and, for shell elements, $u_z = 0$ at all nodes.

Step 1

A distributed pressure of 1000/length is applied on each edge (for shell elements, equivalent concentrated loads). Equivalent concentrated shear forces corresponding to distributed shear loading of 1000/length are applied on each edge in the directions shown.

Response:

Stresses

At every integration point $\sigma_{xx} = \sigma_{yy} = \sigma_{xy} = -1000$ and, for plane strain elements, $\sigma_{zz} = -600$.

Strains

Plane strain elements:

$\varepsilon_{xx} = \varepsilon_{yy} = -1.7333 \times 10^{-5}$, $\gamma_{xy} = -8.6667 \times 10^{-5}$.

Plane stress and shell elements:

$\varepsilon_{xx} = \varepsilon_{yy} = -2.3333 \times 10^{-5}$, $\gamma_{xy} = -8.6667 \times 10^{-5}$.

Displacements

$u_x = x\varepsilon_{xx} + y\gamma_{xy}$, $u_y = y\varepsilon_y$.

For lower-order elements the test description is complete. For higher-order elements another step definition is included.

Step 2

Hydrostatic pressure loading along the two vertical faces, varying from 0 at the top to 1000/length at the bottom, is added to the loads already applied in Step 1.

Response:

Stresses

$\sigma_{xx} = -1000(2 - y)$, $\sigma_{yy} = \sigma_{xy} = -1000$, and, for plane strain elements, $\sigma_{zz} = \nu\left(\sigma_{xx} + \sigma_{yy}\right)$.

Strains

Plane strain elements:

$\varepsilon_{xx} = (-3.0333\ (2 - y) + 1.3) \times 10^{-5}$, $\varepsilon_{yy} = (1.3(2 - y) - 3.03333) \times 10^{-5}$, $\gamma_{xy} = -8.66667 \times 10^{-5}$.

Plane stress and shell elements:

$\varepsilon_{xx} = (-3.333\ (2 - y) + 1) \times 10^{-5}$, $\varepsilon_{yy} = ((2 - y) - 3.3333) \times 10^{-5}$, $\gamma_{xy} = -8.6667 \times 10^{-5}$.

Results and discussion

The results for generalized plane strain elements depend on the boundary constraints applied to the generalized plane strain reference node. In these tests the reference nodes in the lower-order generalized plane strain elements are constrained such that the results are the same as their plane strain counterparts. For the higher-order generalized plane strain elements these nodes are unconstrained, so the results are the same as their plane stress counterparts.

Elements using reduced integration may have additional boundary conditions to those specified above. All elements yield exact solutions.

Input files

ecs3sfs1.inp	CPS3 elements.

ecs4sfs1.inp	CPS4 elements.
ecs4sis1.inp	CPS4I elements.
ecs4srs1.inp	CPS4R elements.
ecs4trs1.inp	CPS4RT elements.
ecs6sfs1.inp	CPS6 elements.
ecs6sks1.inp	CPS6M elements.
ecs6tks1.inp	CPS6MT elements.
ecs8sfs1.inp	CPS8 elements.
ecs8srs1.inp	CPS8R elements.
ece3sfs1.inp	CPE3 elements.
ece3shs1.inp	CPE3H elements.
ece4sfs1.inp	CPE4 elements.
ece4shs1.inp	CPE4H elements.
ece4sis1.inp	CPE4I elements.
ece4sjs1.inp	CPE4IH elements.
ece4srs1.inp	CPE4R elements.
ece4sys1.inp	CPE4RH elements.
ece4tys1.inp	CPE4RHT elements.
ece4trs1.inp	CPE4RT elements.
ece6sfs1.inp	CPE6 elements.
ece6shs1.inp	CPE6H elements.
ece6sks1.inp	CPE6M elements.
ece6sls1.inp	CPE6MH elements.
ece6tls1.inp	CPE6MHT elements.
ece8sfs1.inp	CPE8 elements.
ece8shs1.inp	CPE8H elements.
ece8srs1.inp	CPE8R elements.
ece8sys1.inp	CPE8RH elements.
ecg3sfs1.inp	CPEG3 elements.
ecg3shs1.inp	CPEG3H elements.
ecg4sfs1.inp	CPEG4 elements.
ecg4shs1.inp	CPEG4H elements.
ecg4sis1.inp	CPEG4I elements.
ecg4sjs1.inp	CPEG4IH elements.
ecg4srs1.inp	CPEG4R elements.
ecg4sys1.inp	CPEG4RH elements.
ecg6sfs1.inp	CPEG6 elements.
ecg6shs1.inp	CPEG6H elements.
ecg6sks1.inp	CPEG6M elements.
ecg6sls1.inp	CPEG6MH elements.
ecg8sfs1.inp	CPEG8 elements.
ecg8shs1.inp	CPEG8H elements.
ecg8srs1.inp	CPEG8R elements.

ecg8sys1.inp	CPEG8RH elements.
em33sfs1.inp	M3D3 elements.
em34sfs1.inp	M3D4 elements.
em34srs1.inp	M3D4R elements.
em36sfs1.inp	M3D6 elements.
em38sfs1.inp	M3D8 elements.
em38srs1.inp	M3D8R elements.
em39sfs1.inp	M3D9 elements.
em39srs1.inp	M3D9R elements.
ese4sxs1.inp	S4 elements.
esf4sxs1.inp	S4R elements.
es54sxs1.inp	S4R5 elements.
es68sxs1.inp	S8R elements.
es58sxs1.inp	S8R5 elements.
es59sxs1.inp	S9R5 elements.
es63sxs1.inp	STRI3 elements.
es56sxs1.inp	STRI65 elements.
esc8sxs1.inp	SC8R elements.
esc8sxs1_eh.inp	SC8R elements with enhanced hourglass control.
ecss8sxs1.inp	CSS8 elements.

Generalized plane strain elements with relative motion of bounding planes

Products: Abaqus/Standard

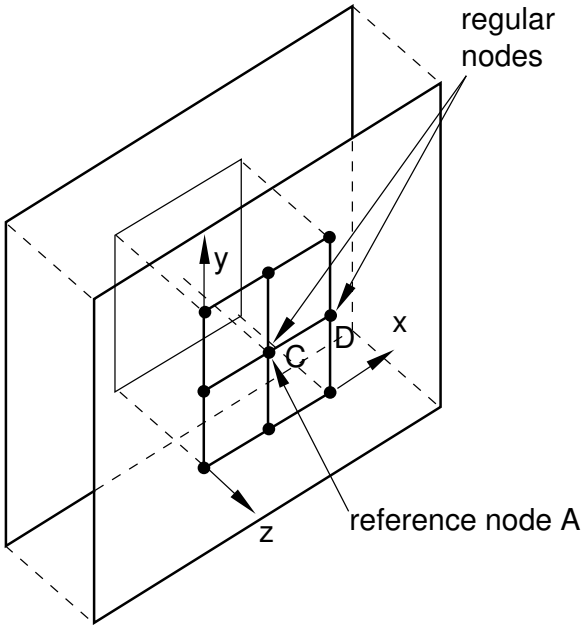
Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

CPEG3 CPEG3H CPEG3HT CPEG3T CPEG4 CPEG4H CPEG4HT CPEG4I CPEG4IH CPEG4R CPEG4RH CPEG4RHT CPEG4RT CPEG4T CPEG6 CPEG6H CPEG6M CPEG6MH CPEG6MHT CPEG6MT CPEG8 CPEG8H CPEG8HT CPEG8R CPEG8RH CPEG8RHT CPEG8T

Problem description



Material: Linear elastic, Young's modulus = 30×10^6, Poisson's ratio = 0.3.

Boundary conditions: $u_x^C = u_y^C = 0, u_y^D = 0$.

Step 1 (Perturbation)

An out-of-plane displacement of 0.01 units (motion of one bounding plane relative to the other) is applied to degree of freedom 3 of the reference node, which is the change in fiber length degree of freedom.

$$u_z^A = 0.01$$
$$\phi_x^A = 0.0$$
$$\phi_y^A = 0.0$$

Analytical solution:

Stresses

At every node $\sigma_{zz} = 3.0 \times 10^5$.

Strains

At every node $\varepsilon_{xx} = \varepsilon_{yy} = -3.0 \times 10^{-3}$, $\varepsilon_{zz} = 1.0 \times 10^{-2}$.

Step 2 (Perturbation)

A relative rotation of 0.01 radians about the *y*-axis is applied to degree of freedom 5 of the reference node (the rotation degree of freedom of one bounding plane relative to the other).

$$u_z^A = 0.0$$
$$\phi_x^A = 0.0$$
$$\phi_y^A = 0.01$$

Analytical solution:

Stresses

Maximum tensile stress $\sigma_{zz}^{max} = 1.5 \times 10^5$.

Strains

Maximum tensile strain $\varepsilon_{zz}^{max} = 5 \times 10^{-3}$.

Results and discussion

For Step 1, all element types yield the exact solution. The results for Step 2 are given in the following table:

Element type	σ_{zz}^{max}	ε_{zz}^{max}
CPEG3	1.264×10^5	4.167×10^{-3}
CPEG3H	1.264×10^5	4.167×10^{-3}
CPEG3HT	1.264×10^5	4.167×10^{-3}
CPEG3T	1.264×10^5	4.167×10^{-3}
CPEG4	1.131×10^5	3.750×10^{-3}
CPEG4H	1.131×10^5	3.750×10^{-3}
CPEG4HT	1.131×10^5	3.750×10^{-3}
CPEG4I	1.500×10^5	5.000×10^{-3}
CPEG4IH	1.500×10^5	5.000×10^{-3}
CPEG4R	1.125×10^5	3.750×10^{-3}

Element type	σ_{zz}^{max}	ε_{zz}^{max}
CPEG4RH	1.125×10^{5}	3.750×10^{-2}
CPEG4RHT	1.125×10^{5}	3.750×10^{-2}
CPEG4RT	1.125×10^{5}	3.750×10^{-3}
CPEG4T	1.131×10^{5}	3.750×10^{-3}
CPEG6	1.500×10^{5}	5.000×10^{-3}
CPEG6H	1.500×10^{5}	5.000×10^{-3}
CPEG6M	1.504×10^{5}	5.000×10^{-3}
CPEG6MH	1.504×10^{5}	5.000×10^{-3}
CPEG6MHT	1.504×10^{5}	5.000×10^{-3}
CPEG6MT	1.504×10^{5}	5.000×10^{-3}
CPEG8	1.500×10^{5}	5.000×10^{-3}
CPEG8H	1.500×10^{5}	5.000×10^{-3}
CPEG8HT	1.500×10^{5}	5.000×10^{-3}
CPEG8R	1.500×10^{5}	5.000×10^{-3}
CPEG8RH	1.500×10^{5}	5.000×10^{-3}
CPEG8RHT	1.500×10^{5}	5.000×10^{-3}
CPEG8T	1.500×10^{5}	5.000×10^{-3}

Second-order quadrilateral elements, first-order incompatible mode elements, and quadratic triangles yield the exact solutions. Modified triangles yield nearly exact solutions. Other element types exhibit stiff response.

Input files

ecg3sas2.inp	CPEG3 and CPEG3H elements.
ecg4sas2.inp	CPEG4, CPEG4I, CPEG4R, CPEG4IH, CPEG4H, and CPEG4RH elements.
ecg6sas2.inp	CPEG6, CPEG6H, CPEG6M, and CPEG6MH elements.
ecg8sas2.inp	CPEG8, CPEG8R, CPEG8H, and CPEG8RH elements.
ecg3tas2.inp	CPEG3HT and CPEG3T elements.
ecg4tas2.inp	CPEG4HT, CPEG4RHT, CPEG4RT, and CPEG4T elements.
ecg6tas2.inp	CPEG6, CPEG6H, CPEG6MT, and CPEG6MHT elements.
ecg8tas2.inp	CPEG8HT, CPEG8RHT, and CPEG8T elements.

Three-dimensional solid elements

Products: Abaqus/Standard

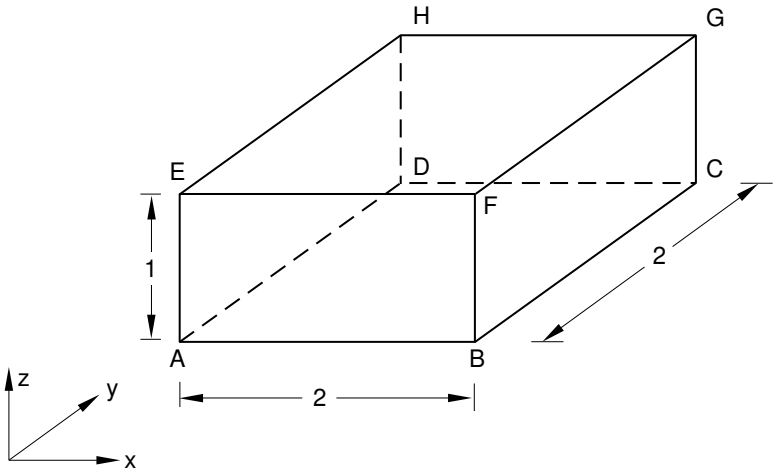
Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D4 C3D4H C3D5 C3D5H C3D6 C3D6H C3D8 C3D8H C3D8I C3D8IH C3D8R C3D8RH C3D10 C3D10H C3D10HS C3D10M C3D10MH C3D15 C3D15H C3D15V C3D15VH C3D20 C3D20H C3D20R C3D20RH C3D27 C3D27H C3D27R C3D27RH CSS8

Problem description



Material: Linear elastic, Young's modulus = 30×10^6, Poisson's ratio = 0.3.

Boundary conditions: $u_x^A = u_y^A = u_z^A = 0$, $u_y^B = 0$, $u_z^D = 0$, $u_x^E = 0$.

Step 1

A distributed pressure of 1000/area is applied on each face, and equivalent concentrated forces for shear loading, defined such that all three shear stresses are of magnitude −1000.

Response:

Stresses

$\sigma_{xx} = \sigma_{yy} = \sigma_{zz} = \sigma_{xy} = \sigma_{xz} = \sigma_{yz} =$ −1000 at every integration point.

Strains

$$\varepsilon_{xx} = \varepsilon_{yy} = \varepsilon_{zz} = -1.3333 \times 10^{-5},\ \gamma_{xy} = \gamma_{yz} = \gamma_{xz} = -8.6667 \times 10^{-5}.$$

Displacements

$$u_x = x\varepsilon_{xx} + y\gamma_{xy}\ ,\quad u_y = y\varepsilon_{yy} + z\gamma_{yz}\ ,\quad u_z = z\varepsilon_{zz} + x\gamma_{xz}.$$

For lower-order elements the test description is complete. For higher-order elements another step definition is included.

Step 2

Hydrostatic pressure loading is applied to the four vertical faces, varying from 0 at top to 1000/area at bottom, in addition to the Step 1 loads.

Response:

Stresses

$$\sigma_{xx} = \sigma_{yy} = -1000(2 - z),\ \sigma_{zz} = -1000,\ \sigma_{xy} = \sigma_{xz} = \sigma_{yz} = -1000.$$

Strains

$$\varepsilon_{xx} = \varepsilon_{yy} = 3.333 \times 10^{-5}(0.7z - 1.1),\ \varepsilon_{zz} = 3.333 \times 10^{-5}(0.2 - 0.6z),$$

$$\gamma_{xy} = \gamma_{xz} = \gamma_{yz} = -8.66667 \times 10^{-5}.$$

Results and discussion

Elements using reduced integration may have additional boundary conditions to those specified above.

Elements C3D27R and C3D27RH employ 21 nodes in this test to produce the exact solutions. The lack of midface nodes is consistent with the elements' intended use, since no contact elements are present.

All elements that do not use the modified formulation, except C3D20RH, yield exact solutions. The stresses calculated for element C3D20RH are correct.

The modified tetrahedral element formulation cannot exactly capture a linearly varying gradient field due to the piecewise linear interpolation used for the unknown field. However, the numerical solution will converge to the exact solution as the mesh is refined.

Section output requests to the results (`.fil`) file and to the data (`.dat`) file are used in some of the input files to output accumulated quantities on the face in the y-z plane. The area of the face is 2.0 in both steps. The accumulated force is reported in a coordinate system that is local to the section. In Step 1 the force is 2000 in each local direction. In Step 2 the total force component in the local 1-direction (normal to the face) changes to 3000.

Input files

ec34sfs2.inp	C3D4 elements.
ec34shs2.inp	C3D4H elements.
ec35sfs2.inp	C3D5 elements.
ec35shs2.inp	C3D5H elements.

ec36sfs2.inp	C3D6 elements.
ec36shs2.inp	C3D6H elements.
ec38sfs2.inp	C3D8 elements.
ec38shs2.inp	C3D8H elements.
ec38sis2.inp	C3D8I elements.
ec38sjs2.inp	C3D8IH elements.
ec38srs2.inp	C3D8R elements.
ec38sys2.inp	C3D8RH elements.
ec3asfs2.inp	C3D10 elements.
ec3ashs2.inp	C3D10H elements.
ec3asis2.inp	C3D10HS elements.
ec3asks2.inp	C3D10M elements.
ec3asls2.inp	C3D10MH elements.
ec3fsfs2.inp	C3D15 elements.
ec3fshs2.inp	C3D15H elements.
ec3isfs2.inp	C3D15V elements.
ec3ishs2.inp	C3D15VH elements.
ec3ksfs2.inp	C3D20 elements.
ec3kshs2.inp	C3D20H elements.
ec3ksrs2.inp	C3D20R elements.
ec3ksys2.inp	C3D20RH elements.
ec3rsfs2.inp	C3D27 elements.
ec3rshs2.inp	C3D27H elements.
ec3rsrs2.inp	C3D27R elements.
ec3rsys2.inp	C3D27RH elements.
ecss8sis2.inp	CSS8 elements.

Axisymmetric solid elements

Products: Abaqus/Standard

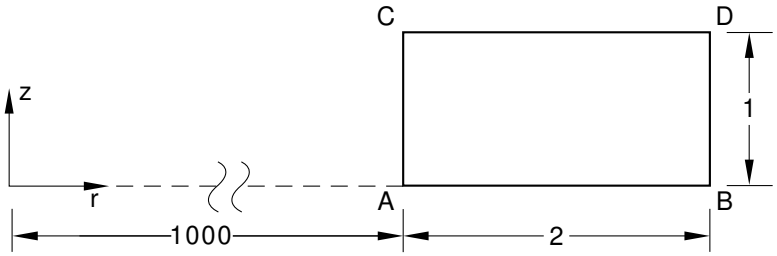
Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

CAX3 CAX3H CAX4 CAX4H CAX4I CAX4IH CAX4R CAX4RH CAX4RHT CAX4RT CAX6 CAX6H CAX6M CAX6MH CAX6MHT CAX6MT CAX8 CAX8H CAX8R CAX8RH

Problem description



Material: Linear elastic, Young's modulus = 30×10^6, Poisson's ratio = 0.3.

For coupled temperature-displacement elements dummy thermal properties are prescribed to complete the material definition.

Boundary conditions: $u_z^A = u_z^B = 0$.

Step 1

A distributed pressure loading of 1000/area is applied on each face.

Response:

Stresses

At every integration point, $\sigma_{rr} = \sigma_{\theta\theta} = \sigma_{zz} = -1000$, $\sigma_{rz} = 0$.

Strains

$\varepsilon_{rr} = \varepsilon_{\theta\theta} = \varepsilon_{zz} = -1.3333 \times 10^{-5}$, $\varepsilon_{rz} = 0$.

Displacements

$u_r = -1.33 \times 10^{-2}$ along $r = 1000$, $u_z = -1.33 \times 10^{-5} z$.

For lower-order elements the test description is complete. For higher-order elements, another step definition is included.

Step 2

Hydrostatic pressure loading is applied along the two vertical faces, varying from 0 at the top to 1000/area at the bottom, in addition to the loads of Step 1.

The following reference solution is obtained for Step 2 using CAXA84 axisymmetric solid elements with nonlinear, asymmetric deformation (input file *eref84s3.inp*) and is given at $z = 0.5$.

Stresses

$\sigma_{rr} = -1500$, $\sigma_{zz} = -1000$, $\sigma_{\theta\theta} = -1500$, $\sigma_{rz} = 0$.

Strains

$\varepsilon_{rr} = -2.5 \times 10^{-5}$, $\varepsilon_{zz} = -3.33 \times 10^{-6}$, $\varepsilon_{\theta\theta} = -2.5 \times 10^{-5}$, $\varepsilon_{rz} = 0$.

Results and discussion

Elements using reduced integration may have additional boundary conditions to those specified above. All elements yield exact solutions.

Section output requests to the results (`.fil`) file and to the data (`.dat`) file are used in some of the input files to output accumulated quantities on the face CD. The quantities are reported in a system that is local to the section.

Input files

eca3sfs3.inp	CAX3 elements.
eca3shs3.inp	CAX3H elements.
eca4sfs3.inp	CAX4 elements.
eca4shs3.inp	CAX4H elements.
eca4sis3.inp	CAX4I elements.
eca4sjs3.inp	CAX4IH elements.
eca4srs3.inp	CAX4R elements.
eca4sys3.inp	CAX4RH elements.
eca4tys3.inp	CAX4RHT elements.
eca4trs3.inp	CAX4RT elements.
eca6sfs3.inp	CAX6 elements.
eca6shs3.inp	CAX6H elements.
eca6sks3.inp	CAX6M elements.
eca6sls3.inp	CAX6MH elements.
eca6tls3.inp	CAX6MHT elements.
eca6tks3.inp	CAX6MT elements.
eca8sfs3.inp	CAX8 elements.
eca8shs3.inp	CAX8H elements.
eca8srs3.inp	CAX8R elements.

eca8sys3.inp	CAX8RH elements.

Axisymmetric solid elements with twist

Products: Abaqus/Standard

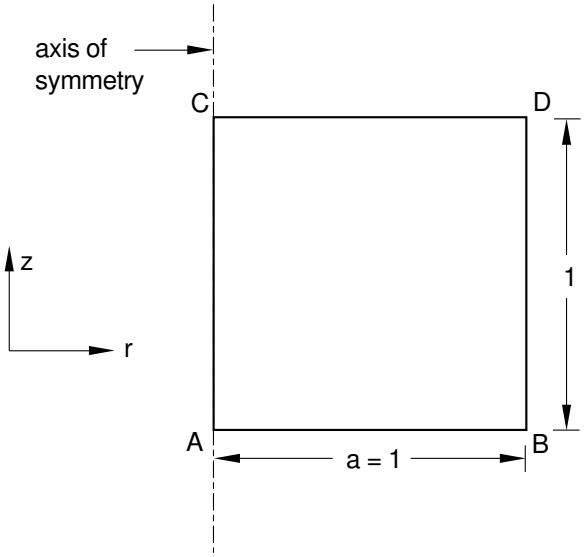
Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

CGAX3 CGAX3H CGAX3HT CGAX3T CGAX4 CGAX4H CGAX4HT CGAX4R CGAX4RH CGAX4RHT CGAX4RT CGAX4T CGAX6 CGAX6H CGAX6M CGAX6MH CGAX6MHT CGAX6MT CGAX8 CGAX8H CGAX8HT CGAX8R CGAX8RH CGAX8RHT CGAX8RT CGAX8T

Problem description



Material: Linear elastic, Young's modulus = 10^6, Poisson's ratio = 0.3.

Boundary conditions: $u_z^A = u_z^B = u_z^C = u_z^D = 0$; $u_r^A = u_r^C = 0$; $\phi^A = \phi^B = 0$.

Step 1

A concentrated moment loading equivalent to a distributed moment loading *M* of 6042 is applied on top face CD.

Analytical solution (L=1):

Twist

$\phi = 2ML/\mu\pi a^4$ = 0.01 (on top face CD).

Stresses

$$\sigma_{z\theta} = \mu\phi r/L.$$

Results and discussion

All elements yield the analytical solution.

Section output requests to the results (`.fil`) file and to the data (`.dat`) file are used in some of the input files to output accumulated quantities on the face CD. The area of the face is 3.142.

Input files

eca3gfs3.inp	CGAX3 elements.
eca3ghs3.inp	CGAX3H elements.
eca3hhs3.inp	CGAX3HT elements.
eca3hfs3.inp	CGAX3T elements.
eca4gfs3.inp	CGAX4 elements.
eca4ghs3.inp	CGAX4H elements.
eca4hhs3.inp	CGAX4HT elements.
eca4grs3.inp	CGAX4R elements.
eca4gys3.inp	CGAX4RH elements.
eca4hys3.inp	CGAX4RHT elements.
eca4hrs3.inp	CGAX4RT elements.
eca4hfs3.inp	CGAX4T elements.
eca6gfs3.inp	CGAX6 elements.
eca6ghs3.inp	CGAX6H elements.
eca6gks3.inp	CGAX6M elements.
eca6gls3.inp	CGAX6MH elements.
eca6hls3.inp	CGAX6MHT elements.
eca6hks3.inp	CGAX6MT elements.
eca8gfs3.inp	CGAX8 elements.
eca8ghs3.inp	CGAX8H elements.
eca8hhs3.inp	CGAX8HT elements.
eca8grs3.inp	CGAX8R elements.
eca8gys3.inp	CGAX8RH elements.
eca8hys3.inp	CGAX8RHT elements.
eca8hrs3.inp	CGAX8RT elements.
eca8hfs3.inp	CGAX8T elements.

Cylindrical elements

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

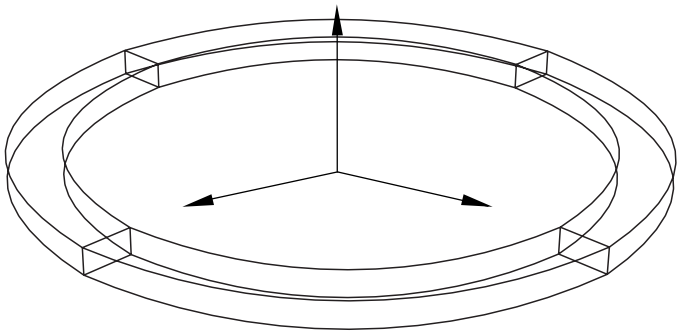
Elements tested

CCL9 CCL9H CCL18 CCL18H CCL12 CCL12H
CCL24 CCL24R CCL24H CCL24RH

Features tested

Elements are tested for concentrated and distributed load cases. Different types of analyses (linear and nonlinear) are studied. Both elastic and hyperelastic material models are used.

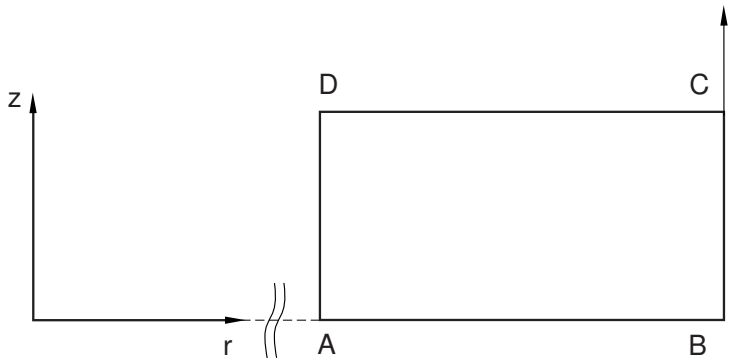
Problem description



Mesh: The mesh presented above is used for elements with a rectangular cross-section. For elements with a triangular cross-section, two elements are used for each element represented above. The axis of symmetry is the z-axis.

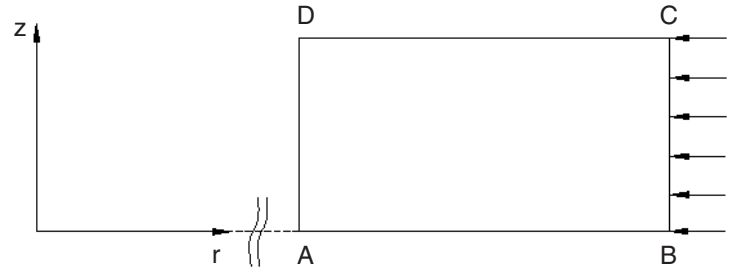
Material: Linear elasticity: Linear elastic, Young's modulus = 30×10^6, Poisson's ratio = 0.3. Hyperelasticity: Hyperelastic, polynomial strain energy potential, N=2, $C_{10} = 1 \times 10^5$, $C_{01} = 0.5 \times 10^5$, $C_{20} = 0.5 \times 10^5$, $C_{11} = 0.8 \times 10^5$, $C_{02} = 0.75 \times 10^5$, $D_1 = 1 \times 10^{-7}$, $D_2 = 1 \times 10^{-7}$.

Boundary conditions: CASE 1



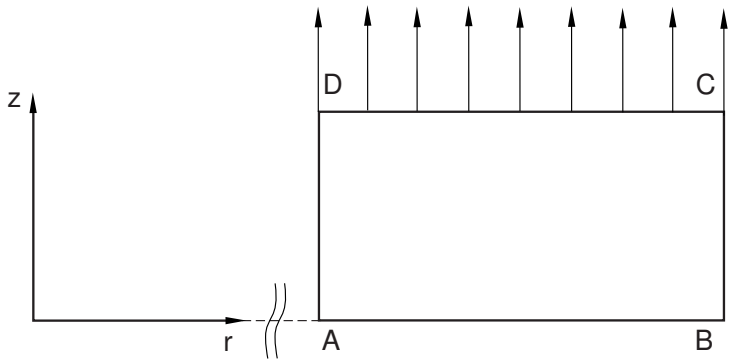
Segment AD is fixed. Axisymmetric boundary conditions are enforced.

CASE 2



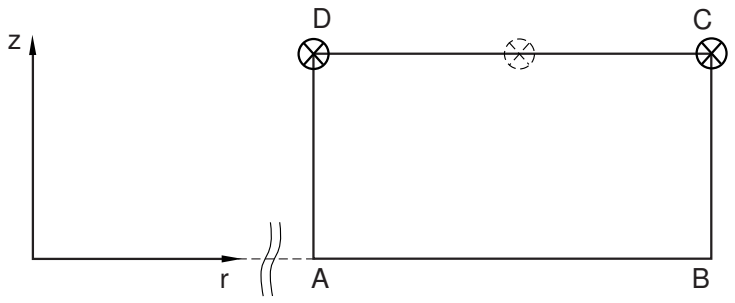
Segment AD is fixed. Axisymmetric boundary conditions are enforced.

CASE 3



CCL12 and CCL9: Segment AD is fixed. Axisymmetric boundary conditions are enforced.
CCL24 and CCL18: Segment AB is fixed. Axisymmetric boundary conditions are enforced.

CASE 4



Segment AB is fixed.

Loading: CASE 1: concentrated load case
CASE 2: distributed load case
CASE 3: distributed surface load case
CASE 4: concentrated load case
(See previous figures)

Results and discussion

The results are compared to the results obtained using axisymmetric elements. CCL9 elements are compared to CAX3 (and CGAX3, when appropriate), CCL12 elements are compared to CAX4 (and CGAX4, when appropriate), CCL18 elements are compared to CAX6 (and CGAX6, when appropriate), and CCL24 are compared to CAX8 (and CGAX8, when appropriate). Cylindrical elements and axisymmetric elements yield the same results with differences less than 2%.

Input files

ecc9gfs1a.inp	CCL9 elements, load and boundary conditions of Case 1, perturbation step, elastic material.
ecc9gfs1b.inp	CCL9 elements, load and boundary conditions of Case 1, general step, elastic material.
ecc9gfs1c.inp	CCL9 elements, load and boundary conditions of Case 1, assumed nonlinear geometry, elastic material.
ecc9ghs1d.inp	CCL9H elements, load and boundary conditions of Case 1, assumed nonlinear geometry, hyperelastic material.
ecc9gfs2a.inp	CCL9 elements, load and boundary conditions of Case 2, perturbation step, elastic material.
ecc9gfs2b.inp	CCL9 elements, load and boundary conditions of Case 2, general step, elastic material.
ecc9gfs2c.inp	CCL9 elements, load and boundary conditions of Case 2, assumed nonlinear geometry, elastic material.
ecc9ghs2d.inp	CCL9H elements, load and boundary conditions of Case 2, assumed nonlinear geometry, hyperelastic material.
ecc9gfs3a.inp	CCL9 elements, load and boundary conditions of Case 3, perturbation step, elastic material.
ecc9gfs3b.inp	CCL9 elements, load and boundary conditions of Case 3, general step, elastic material.

ecc9gfs3c.inp	CCL9 elements, load and boundary conditions of Case 3, assumed nonlinear geometry, elastic material.
ecc9ghs3d.inp	CCL9H elements, load and boundary conditions of Case 3, assumed nonlinear geometry, hyperelastic material.
ecc9gfs4a.inp	CCL9 elements, load and boundary conditions of Case 4, perturbation step, elastic material.
ecc9gfs4b.inp	CCL9 elements, load and boundary conditions of Case 4, general step, elastic material.
ecc9gfs4c.inp	CCL9 elements, load and boundary conditions of Case 4, assumed nonlinear geometry, elastic material.
ecc9ghs4d.inp	CCL9H elements, load and boundary conditions of Case 4, assumed nonlinear geometry, hyperelastic material.
ecccgfs1a.inp	CCL12 elements, load and boundary conditions of Case 1, perturbation step with **LOAD CASE*, elastic material.
ecccgfs1b.inp	CCL12 elements, load and boundary conditions of Case 1, general step, elastic material.
ecccgfs1c.inp	CCL12 elements, load and boundary conditions of Case 1, assumed nonlinear geometry, elastic material.
ecccghs1d.inp	CCL12H elements, load and boundary conditions of Case 1, assumed nonlinear geometry, hyperelastic material.
ecccgfs2a.inp	CCL12 elements, load and boundary conditions of Case 2, perturbation step, elastic material.
ecccgfs2b.inp	CCL12 elements, load and boundary conditions of Case 2, general step, elastic material.
ecccgfs2c.inp	CCL12 elements, load and boundary conditions of Case 2, assumed nonlinear geometry, additional linear perturbation step with **LOAD CASE*, elastic material.
ecccghs2d.inp	CCL12H elements, load and boundary conditions of Case 2, assumed nonlinear geometry, hyperelastic material.
ecccgfs3a.inp	CCL12 elements, load and boundary conditions of Case 3, perturbation step, elastic material.
ecccgfs3b.inp	CCL12 elements, load and boundary conditions of Case 3, general step, elastic material.
ecccgfs3c.inp	CCL12 elements, load and boundary conditions of Case 3, assumed nonlinear geometry, elastic material.
ecccghs3d.inp	CCL12H elements, load and boundary conditions of Case 3, assumed nonlinear geometry, hyperelastic material.
ecccgfs4a.inp	CCL12 elements, load and boundary conditions of Case 4, perturbation step, elastic material.
ecccgfs4b.inp	CCL12 elements, load and boundary conditions of Case 4, general step, elastic material.
ecccgfs4c.inp	CCL12 elements, load and boundary conditions of Case 4, assumed nonlinear geometry, elastic material.
ecccghs4d.inp	CCL12H elements, load and boundary conditions of Case 4, assumed nonlinear geometry, hyperelastic material.
eccigfs1a.inp	CCL18 elements, load and boundary conditions of Case 1, perturbation step, elastic material.
eccigfs1b.inp	CCL18 elements, load and boundary conditions of Case 1, general step, elastic material.

eccigfs1c.inp	CCL18 elements, load and boundary conditions of Case 1, assumed nonlinear geometry, elastic material.
eccighs1d.inp	CCL18H elements, load and boundary conditions of Case 1, assumed nonlinear geometry, hyperelastic material.
eccigfs2a.inp	CCL18 elements, load and boundary conditions of Case 2, perturbation step, elastic material.
eccigfs2b.inp	CCL18 elements, load and boundary conditions of Case 2, general step, elastic material.
eccigfs2c.inp	CCL18 elements, load and boundary conditions of Case 2, assumed nonlinear geometry, elastic material.
eccighs2d.inp	CCL18H elements, load and boundary conditions of Case 2, assumed nonlinear geometry, hyperelastic material.
eccigfs3a.inp	CCL18 elements, load and boundary conditions of Case 3, perturbation step, elastic material.
eccigfs3b.inp	CCL18 elements, load and boundary conditions of Case 3, general step, elastic material.
eccigfs3c.inp	CCL18 elements, load and boundary conditions of Case 3, assumed nonlinear geometry, elastic material.
eccighs3d.inp	CCL18H elements, load and boundary conditions of Case 3, assumed nonlinear geometry, hyperelastic material.
eccigfs4a.inp	CCL18 elements, load and boundary conditions of Case 4, perturbation step, elastic material.
eccigfs4b.inp	CCL18 elements, load and boundary conditions of Case 4, general step, elastic material.
eccigfs4c.inp	CCL18 elements, load and boundary conditions of Case 4, assumed nonlinear geometry, elastic material.
eccighs4d.inp	CCL18H elements, load and boundary conditions of Case 4, assumed nonlinear geometry, hyperelastic material.
eccrgfs1a.inp	CCL24 elements, load and boundary conditions of Case 1, perturbation step, elastic material.
eccrgfs1b.inp	CCL24 elements, load and boundary conditions of Case 1, general step, elastic material.
eccrgfs1c.inp	CCL24 elements, load and boundary conditions of Case 1, assumed nonlinear geometry, elastic material.
eccrghs1d.inp	CCL24H elements, load and boundary conditions of Case 1, assumed nonlinear geometry, hyperelastic material.
eccrgfs2a.inp	CCL24 elements, load and boundary conditions of Case 2, perturbation step, elastic material.
eccrgfs2b.inp	CCL24 elements, load and boundary conditions of Case 2, general step, elastic material.
eccrgfs2c.inp	CCL24 elements, load and boundary conditions of Case 2, assumed nonlinear geometry, elastic material.
eccrghs2d.inp	CCL24H elements, load and boundary conditions of Case 2, assumed nonlinear geometry, hyperelastic material.
eccrgfs3a.inp	CCL24 elements, load and boundary conditions of Case 3, perturbation step, elastic material.
eccrgfs3b.inp	CCL24 elements, load and boundary conditions of Case 3, general step, elastic material.

eccrgfs3c.inp	CCL24 elements, load and boundary conditions of Case 3, assumed nonlinear geometry, elastic material.
eccrghs3d.inp	CCL24H elements, load and boundary conditions of Case 3, assumed nonlinear geometry, hyperelastic material.
eccrgfs4a.inp	CCL24 elements, load and boundary conditions of Case 4, perturbation step, elastic material.
eccrgfs4b.inp	CCL24 elements, load and boundary conditions of Case 4, general step, elastic material.
eccrgfs4c.inp	CCL24 elements, load and boundary conditions of Case 4, assumed nonlinear geometry, elastic material.
eccrghs4d.inp	CCL24H elements, load and boundary conditions of Case 4, assumed nonlinear geometry, hyperelastic material.
eccrgrs1a.inp	CCL24R elements, load and boundary conditions of Case 1, perturbation step, elastic material.
eccrgrs1b.inp	CCL24R elements, load and boundary conditions of Case 1, general step, elastic material.
eccrgrs1c.inp	CCL24R elements, load and boundary conditions of Case 1, assumed nonlinear geometry, elastic material.
eccrgys1d.inp	CCL24RH elements, load and boundary conditions of Case 1, assumed nonlinear geometry, hyperelastic material.
eccrgrs2a.inp	CCL24R elements, load and boundary conditions of Case 2, perturbation step, elastic material.
eccrgrs2b.inp	CCL24R elements, load and boundary conditions of Case 2, general step, elastic material.
eccrgrs2c.inp	CCL24R elements, load and boundary conditions of Case 2, assumed nonlinear geometry, elastic material.
eccrgys2d.inp	CCL24RH elements, load and boundary conditions of Case 2, assumed nonlinear geometry, hyperelastic material.
eccrgrs3a.inp	CCL24R elements, load and boundary conditions of Case 3, perturbation step, elastic material.
eccrgrs3b.inp	CCL24R elements, load and boundary conditions of Case 3, general step, elastic material.
eccrgrs3c.inp	CCL24R elements, load and boundary conditions of Case 3, assumed nonlinear geometry, elastic material.
eccrgys3d.inp	CCL24RH elements, load and boundary conditions of Case 3, assumed nonlinear geometry, hyperelastic material.
eccrgrs4a.inp	CCL24R elements, load and boundary conditions of Case 4, perturbation step, elastic material.
eccrgrs4b.inp	CCL24R elements, load and boundary conditions of Case 4, general step, elastic material.
eccrgrs4c.inp	CCL24R elements, load and boundary conditions of Case 4, assumed nonlinear geometry, elastic material.
eccrgys4d.inp	CCL24RH elements, load and boundary conditions of Case 4, assumed nonlinear geometry, hyperelastic material.

Loading of piezoelectric elements

Products: Abaqus/Standard

Overview

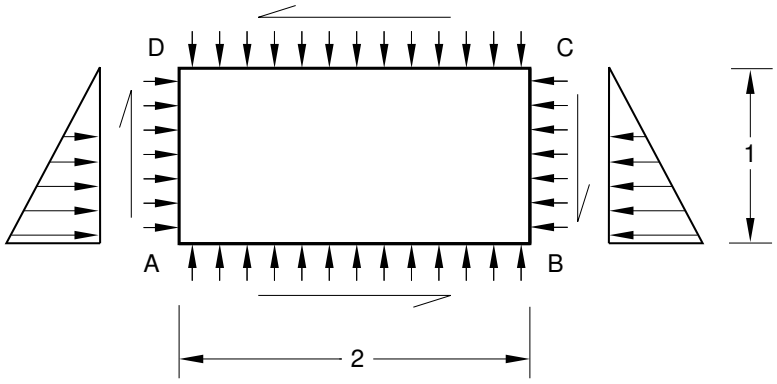
This problem contains basic test cases for one or more Abaqus elements and features.

Plane stress and plane strain piezoelectric elements

Elements tested

CPS3E CPS4E CPS6E CPS8E CPS8RE CPE3E CPE4E CPE6E CPE8E CPE8RE

Problem description



Material: Linear elastic, Young's modulus = 30×10^6, Poisson's ratio = 0.3, no piezoelectric coupling, isotropic dielectric constant 1.0×10^{-3}.

Boundary conditions: $u_x^A = u_y^A = \varphi^A = 0$, $u_y^B = \varphi^B = 0$.

Loading: Distributed pressure of 1000/length on each edge.

Equivalent concentrated shear forces corresponding to distributed shear loading of 1000/length on each edge in the directions shown.

Distributed charges of 1000/length on each edge.

Concentrated charges at each node to negate the distributed charges, except for the distributed charge of 1000/length on the top surface.

Reference solution

Stresses

Both plane stress and plane strain elements,

$\sigma_{xx} = \sigma_{yy} = \sigma_{xy} = -1000$;

and for plane strain elements,

$\sigma_{zz} = -600$.

Strains

Plane strain elements,

$\varepsilon_{xx} = \varepsilon_{yy} = -1.7333 \times 10^{-5}$, $\gamma_{xy} = -8.6667 \times 10^{-5}$.

Plane stress elements,

$\varepsilon_{xx} = \varepsilon_{yy} = -2.3333 \times 10^{-5}$, $\gamma_{xy} = -8.6667 \times 10^{-5}$.

Electrical fluxes

Both plane stress and plane strain elements, $q_{xx} = 0$, $q_{yy} = -1000$.

Electrical potential gradients

Both plane stress and plane strain elements, $E_{xx} = 0$, $E_{yy} = -1.0 \times 10^{6}$.

Displacements

$u_x = x\varepsilon_{xx} + y\gamma_{xy}$, $u_y = y\varepsilon_{yy}$.

Potentials

$\varphi = -yE_{yy}$.

Results and discussion

Elements using reduced integration may have additional boundary conditions to those specified above. All elements yield exact solutions.

Section output requests to the results (`.fil`) file and to the data (`.dat`) file are used in some of the input files to output accumulated quantities on the face in the *x*–*y* plane.

Input files

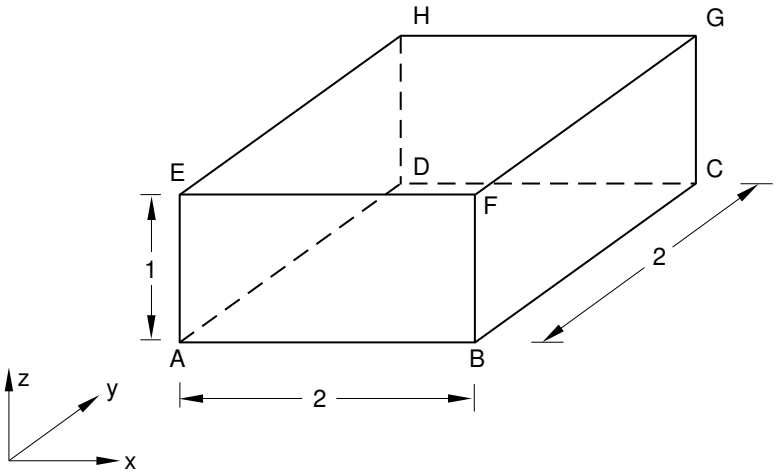
ecs3efs1.inp	CPS3E elements.
ecs4efs1.inp	CPS4E elements.
ecs6efs1.inp	CPS6E elements.
ecs8efs1.inp	CPS8E elements.
ecs8ers1.inp	CPS8RE elements.
ece3efs1.inp	CPE3E elements.
ece4efs1.inp	CPE4E elements.
ece6efs1.inp	CPE6E elements.
ece8efs1.inp	CPE8E elements.
ece8ers1.inp	CPE8RE elements.

Three-dimensional piezoelectric elements

Elements tested

C3D4E C3D6E C3D8E C3D10E C3D15E C3D20E C3D20RE

Problem description



Material: Linear elastic, Young's modulus 30×10^6, Poisson's ratio 0.3, no piezoelectric coupling, isotropic dielectric constant 1.0×10^{-3}.

Boundary conditions: $u_x^A = u_y^A = u_z^A = \varphi^A = 0$, $u_y^B = \varphi^B = 0$, $\varphi^C = 0$, $u_z^D = \varphi^D = 0$, $u_x^E = 0$.

Loading: Distributed pressure of 1000/area on each face, and equivalent concentrated forces for shear loading, defined such that all three shear stresses are of magnitude −1000.

Distributed charges of 1000/area on each face.

Concentrated charges at each node to negate the distributed charges, except for the distributed charge of 1000/area on the top surface.

Reference solution

Stresses

$$\sigma_{xx} = \sigma_{yy} = \sigma_{xy} = \sigma_{xy} = \sigma_{xz} = \sigma_{yz} = -1000.$$

Strains

$$\varepsilon_{xx} = \varepsilon_{yy} = \varepsilon_{zz} = -1.3333 \times 10^{-5},\ \gamma_{xy} = \gamma_{yz} = \gamma_{xz} = -8.6667 \times 10^{-5}.$$

Electrical fluxes

$$q_{xx} = 0,\ q_{yy} = 0,\ q_{zz} = -1000.$$

Electrical potential gradients

$E_{xx} = 0$, $E_{yy} = 0$, $E_{zz} = -1.0 \times 10^6$.

Displacements

$u_x = x\varepsilon_{xx} + y\gamma_{xy}, u_y = y\varepsilon_{yy} + z\gamma_{yz}, u_z = z\varepsilon_{zz} + x\gamma_{xz}$.

Potentials

$\varphi = -zE_{zz}$.

Results and discussion

Elements using reduced integration may have additional boundary conditions to those specified above. All elements yield exact solutions.

Section output requests to the results (`.fil`) file and to the data (`.dat`) file are used in some of the input files to output accumulated quantities on the face in the *x*–*y* plane.

Input files

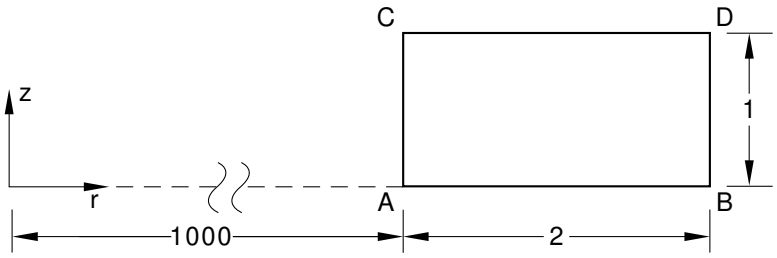
ec34efs2.inp	C3D4E elements.
ec36efs2.inp	C3D6E elements.
ec38efs2.inp	C3D8E elements.
ec3aefs2.inp	C3D10E elements.
ec3fefs2.inp	C3D15E elements.
ec3kefs2.inp	C3D20E elements.
ec3kers2.inp	C3D20RE elements.

Axisymmetric piezoelectric elements

Elements tested

CAX3E CAX4E CAX6E CAX8E CAX8RE

Problem description



Material: Linear elastic, Young's modulus 30×10^6, Poisson's ratio 0.3, no piezoelectric coupling, isotropic dielectric constant 1.0×10^{-3}.

Boundary conditions: $u_z^A = u_z^B = \varphi^A = \varphi^B = 0$.

Loading: Distributed pressure of 1000/area on each face.

Distributed charges of 1000/area on each face.

Concentrated charges at each node to negate the distributed charges, except for the distributed charge of 1000/area on the top surface.

Reference solution

Stresses

$\sigma_{rr} = \sigma_{\theta\theta} = \sigma_{zz} = -1000$, $\sigma_{rz} = 0$.

Strains

$\varepsilon_{rr} = \varepsilon_{\theta\theta} = \varepsilon_{zz} = -1.3333 \times 10^{-5}$, $\varepsilon_{rz} = 0$.

Electrical fluxes

$q_{rr} = 0$, $q_{zz} = -1000$.

Electrical potential gradients

$E_{rr} = 0$, $E_{zz} = -1.0 \times 10^6$.

Displacements

$u_r = -1.33 \times 10^{-2}$ along $r = 1000$, $u_z = -1.33 \times 10^{-5} z$.

Potentials

$\varphi = -zE_{zz}$.

Results and discussion

Elements using reduced integration may have additional boundary conditions to those specified above. All elements yield exact solutions.

Section output requests to the results (`.fil`) file and to the data (`.dat`) file are used in some of the input files to output accumulated quantities on the face in the *x*–*y* plane.

Input files

eca3efs3.inp	CAX3E elements.
eca4efs3.inp	CAX4E elements.
eca6efs3.inp	CAX6E elements.

eca8efs3.inp	CAX8E elements.
eca8ers3.inp	CAX8RE elements.

Love-Kirchhoff beams and shells

Products: Abaqus/Standard

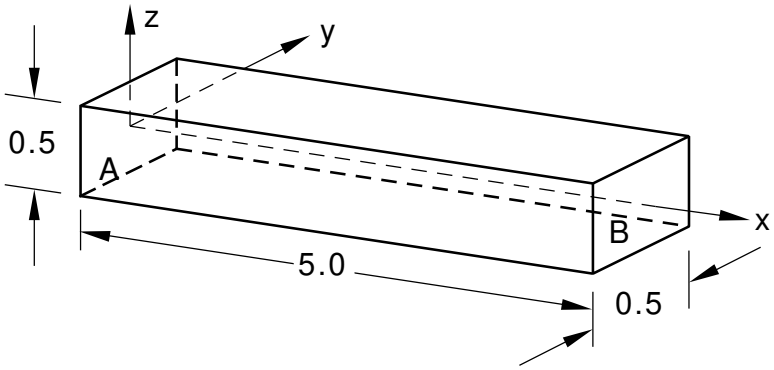
Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

B23 B23H B33 B33H STRI3 STRI65

Problem description



A three-dimensional problem is shown here. It may be particularized for two-dimensional beam elements.

Material: Linear elastic, Young's modulus = 30×10^6, Poisson's ratio = 0.3.

Boundary conditions: $\phi_x = \phi_y = \phi_z = 0$ at end A, $u_x = u_y = u_z = 0$ at end B.

Loading: $F_x = F_y = F_z =$ 25.0 at end A, $M_x = M_y = M_z =$ 100.0 at end B. Only F_x, F_z, and M_y are applied for shell models.

General: Gauss integration is used for the shell cross-section for element STRI3.

Reference solution

Displacements in beam elements

$u_x = 1.667 \times 10^{-5}$, $u_y = -1.333 \times 10^{-3}$, $u_z = 0.01467$ at node A,

$\phi_x = 4.92 \times 10^{-3}$, $\phi_y = 5.2 \times 10^{-3}$, $\phi_z = 1.2 \times 10^{-3}$ at node B.

Stress resultants in beam elements

$N_x = -25.0$, $M_{11} = 25(1 - x)$, $M_{22} = 100 + 25(5 - x)$, $M_x = 100.0$.

Displacements in shell elements

$u_x = 1.667 \times 10^{-5}$, $u_z = 0.01467$ at node A, $\phi_y = 5.2 \times 10^{-3}$ at node B.

Results and discussion

Beam elements yield exact solutions. 3-node shell elements yield exact solutions for u_x and ϕ_y but yield a value of 0.01412 for u_z. 6-node shell elements yield exact solutions for u_x and ϕ_y but yield a value of 0.01464 for u_z.

Input files

eb2arxs4.inp	B23 elements.
eb2jrxs4.inp	B23H elements.
eb3arxs4.inp	B33 elements.
eb3jrxs4.inp	B33H elements.
es63sxs4.inp	STRI3 elements.
es56sxs4.inp	STRI65 elements.

Shear flexible beams and shells: I

Products: Abaqus/Standard Abaqus/Explicit

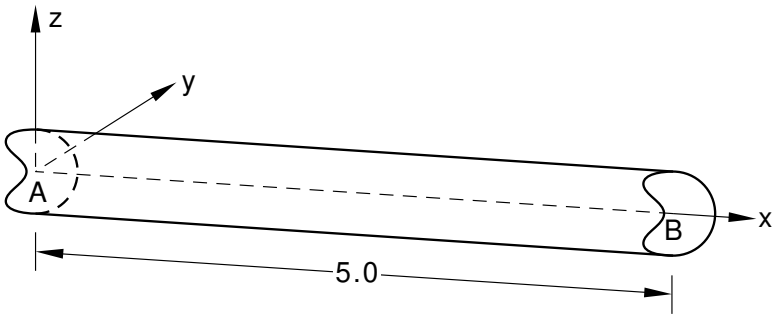
Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

B21 B21H B22 B22H B31 B31H B31OS B31OSH B32 B32H B32OS B32OSH
PIPE21 PIPE21H PIPE22 PIPE22H PIPE31 PIPE31H PIPE32 PIPE32H
S4 S4R S4R5 S8R S8R5 S9R5

Problem description



A three-dimensional problem is shown here, which can be particularized for two-dimensional beam elements.

Material: Linear elastic, Young's modulus = 30×10^6, Poisson's ratio = 0.3.

Boundary conditions: $\phi_x = \phi_y = \phi_z = 0$ at end A, $u_x = u_y = u_z = 0$ at end B.

Loading: $F_x = F_y = F_z =$ 25.0 at end A. Only F_x and F_z are applied for shell models.

Section properties: $A =$ 0.25, $I_{11} = I_{22} =$ 1×10^6, $J =$ 0.0104167. The bending inertias have intentionally been chosen as very large values in order to test the shear-only modes.

For pipe elements a circular cross-section of outer radius 0.5 and wall thickness 0.05 is used. For this case a different analytical solution based upon Timoshenko theory is used for comparison.

Analogous problems are modeled in Abaqus/Explicit using linear beam and pipe elements. Unit density is prescribed for the material, and the solution is computed for unit time. Loads are applied smoothly for a quasi-static solution, similar to that from static analysis. The results using pipe elements are consistent to that using beam elements, both of which match the static analysis.

Reference solution

Displacements in beam elements

$u_x = \frac{PL}{EA}$, $u_y = u_z = \frac{PL^3}{3EI}\left(1 + \frac{3EI}{kGAL^2}\right)$ at node A.

Regular and open section elements

$u_x = 1.667 \times 10^{-5}$, $u_y = u_z = 4.333 \times 10^{-5}$.

Pipe elements

$u_x = 2.792 \times 10^{-5}$, $u_y = u_z = 2.194 \times 10^{-3}$.

Stress resultants in beam and pipe elements

$SF1 = -25.0$, $SM1 = 25(5 - x)$, $SM2 = 25(5 - x)$,

Transverse shear: $SF3 = 25.0$, $SF2 = -25.0$.

Displacements in shell elements

$u_x = 1.667 \times 10^{-5}$, $u_z = 4.333 \times 10^{-5}$ at node A.

Results and discussion

All beam and shell elements yield exact solutions. Pipe element solutions are given in *Table 1*.

Table 1: Pipe element solutions.

	u_x	$u_y = u_z$
Analytical solution	2.792×10^{-5}	2.194×10^{-3}
Linear pipe elements	2.792×10^{-5}	2.093×10^{-3}
Quadratic pipe elements	2.792×10^{-5}	2.106×10^{-3}

Input files

eb22gxs5.inp	B21 elements.
eb2hgxs5.inp	B21H elements.
eb23gxs5.inp	B22 elements.
eb2igxs5.inp	B22H elements.
eb32gxs5.inp	B31 elements.
eb3hgxs5.inp	B31H elements.
ebo2gxs5.inp	B31OS elements.
ebohgxs5.inp	B31OSH elements.
eb33gxs5.inp	B32 elements.

eb3igxs5.inp	B32H elements.
ebo3gxs5.inp	B32OS elements.
eboigxs5.inp	B32OSH elements.
ep22pxs5.inp	PIPE21 elements.
ep2hpxs5.inp	PIPE21H elements.
ep23pxs5.inp	PIPE22 elements.
ep2ipxs5.inp	PIPE22H elements.
ep32pxs5.inp	PIPE31 elements.
ep3hpxs5.inp	PIPE31H elements.
ep33pxs5.inp	PIPE32 elements.
ep3ipxs5.inp	PIPE32H elements.
ese4sgs5.inp	S4 elements.
esf4sgs5.inp	S4R elements.
es54sgs5.inp	S4R5 elements.
es68sgs5.inp	S8R elements.
es58sgs5.inp	S8R5 elements.
es59sgs5.inp	S9R5 elements.
es56sgs5.inp	STRI65 elements.
force_shearflex_beam2d_xpl.inp	B21 elements in Abaqus/Explicit.
force_shearflex_beam3d_xpl.inp	B31 elements in Abaqus/Explicit.
force_shearflex_pipe2d_xpl.inp	PIPE21 elements in Abaqus/Explicit.
force_shearflex_pipe3d_xpl.inp	PIPE31 elements in Abaqus/Explicit.

Shear flexible beams and shells: II

Products: Abaqus/Standard Abaqus/Explicit

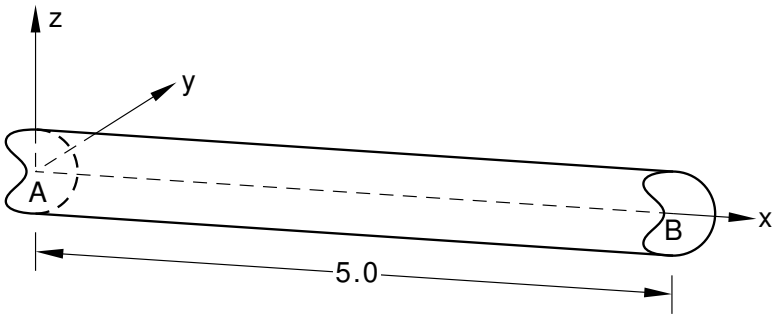
Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

B21 B21H B22 B22H B31 B31H B31OS B31OSH B32 B32H B32OS B32OSH
PIPE21 PIPE21H PIPE22 PIPE22H PIPE31 PIPE31H PIPE32 PIPE32H
S4 S4R S4R5 S8R S8R5 S9R5 STRI65

Problem description



A three-dimensional problem is shown here, which can be particularized for two-dimensional beam elements.

Material: Linear elastic, Young's modulus = 30×10^6, Poisson's ratio = 0.3.

Boundary conditions: $\phi_x = \phi_y = \phi_z = 0$ at end A, $u_x = u_y = u_z = 0$ at end B.

Loading: $M_x = M_y = M_z =$ 100.0 at end B. Only M_y is applied for shell models.

General: Analogous problems are modeled in Abaqus/Explicit using linear beam and pipe elements. Unit density is prescribed for the material, and the solution is computed for unit time. Loads are applied smoothly for a quasi-static solution, similar to that from static analysis. The results using pipe elements are consistent to that using beam elements, both of which match the static analysis.

Reference solution

Displacements in regular beam elements

$u_y = -8 \times 10^{-3}$, $u_z = 8 \times 10^{-3}$ at end A,

$\phi_x = 4.92 \times 10^{-3}$, $\phi_y = 3.2 \times 10^{-3}$, $\phi_z = 3.2 \times 10^{-3}$ at end B.

Displacements in open section beam elements

$u_y = -6.02 \times 10^{-2}$, $u_z = .2$ at end A,

$\phi_y = 8.0 \times 10^{-2}$, $\phi_z = 2.41 \times 10^{-2}$ at end B.

Displacements in pipe elements

$u_y = -2.47 \times 10^{-3}$, $u_z = 2.47 \times 10^{-3}$ at end A,

$\phi_x = 1.28 \times 10^{-3}$, $\phi_y = 9.87 \times 10^{-4}$, $\phi_z = 9.87 \times 10^{-4}$ at end B.

Stress resultants in beam and pipe elements

$N_x = 0.0$, $M_x = 100$, $M_{11} = -100$, $M_{22} = 100$.

Transverse shear = 0.0.

Displacements in shell elements

$u_z = 8 \times 10^{-3}$ at node A, $\phi_y = 3.2 \times 10^{-3}$ at node B.

Results and discussion

All beam and shell elements yield exact solutions. Pipe elements yield the following solutions:

$u_y = -2.475 \times 10^{-3}$, $u_z = 2.475 \times 10^{-3}$ at end A,

$\phi_x = 1.287 \times 10^{-3}$, $\phi_y = 9.90 \times 10^{-4}$, $\phi_z = 9.90 \times 10^{-4}$ at end B.

Input files

eb22rxs6.inp	B21 elements.
eb2hrxs6.inp	B21H elements.
eb23rxs6.inp	B22 elements.
eb2irxs6.inp	B22H elements.
eb32rxs6.inp	B31 elements.
eb32rxs7.inp	B31 elements with a nondefault value of slenderness compensation factor.
eb32rxs8.inp	B31 elements with an internally calculated slenderness compensation factor.
eb3hrxs6.inp	B31H elements.
ebo2ixs6.inp	B31OS elements.
ebohixs6.inp	B31OSH elements.
eb33rxs6.inp	B32 elements.
eb3irxs6.inp	B32H elements.
ebo3ixs6.inp	B32OS elements.
eboiixs6.inp	B32OSH elements.

ep22pxs6.inp	PIPE21 elements.
ep2hpxs6.inp	PIPE21H elements.
ep23pxs6.inp	PIPE22 elements.
ep2ipxs6.inp	PIPE22H elements.
ep32pxs6.inp	PIPE31 elements.
ep3hpxs6.inp	PIPE31H elements.
ep33pxs6.inp	PIPE32 elements.
ep3ipxs6.inp	PIPE32H elements.
ese4sxs6.inp	S4 elements.
esf4sxs6.inp	S4R elements.
es54sxs6.inp	S4R5 elements.
es68sxs6.inp	S8R elements.
es58sxs6.inp	S8R5 elements.
es59sxs6.inp	S9R5 elements.
es56sxs6.inp	STRI65 elements.
moment_shearflex_beam2d_xpl.inp	B21 elements in Abaqus/Explicit.
moment_shearflex_beam3d_xpl.inp	B31 elements in Abaqus/Explicit.
moment_shearflex_pipe2d_xpl.inp	PIPE21 elements in Abaqus/Explicit.
moment_shearflex_pipe3d_xpl.inp	PIPE31 elements in Abaqus/Explicit.

Initial curvature of beams and shells

Products: Abaqus/Standard Abaqus/Explicit

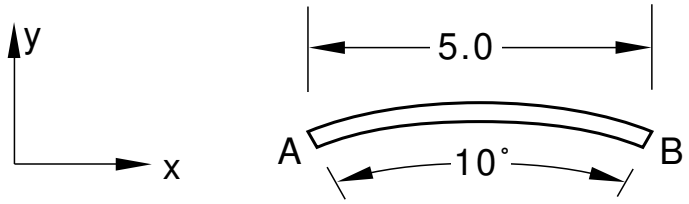
Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

B21 B21H B22 B22H B23 B23H B31 B31H B31OS B31OSH B32 B32H B32OS B32OSH B33 B33H
PIPE21 PIPE21H PIPE22 PIPE22H PIPE31 PIPE31H PIPE32 PIPE32H
S4 S4R S4R5 S8R S8R5 S9R5 STRI3 STRI65 SC6R SC8R CSS8

Problem description



Material: Linear elastic, Young's modulus = 30×10^6, Poisson's ratio = 0.3.

Boundary conditions: End *A* is clamped.

Loading: $F_x =$ 25.0 at end *B*.

General: Initial curvature is defined by specifying the direction cosines of the normals at the two ends.

Gauss integration is used for the shell cross-section for the S4R elements.

Reference solution

Reference results are generated from models consisting of 20 B33 cubic beam elements. (Since only one element is used for modeling, if the direction cosines of the normals are not used, the solution will correspond to straight beam theory.) The reference tests use SECTION=RECT, SECTION=I, or SECTION=PIPE. These sections correspond to regular beams and shells, open section beams, and pipes, respectively. Only pipe elements are verified in Abaqus/Explicit.

Regular beams and shells (see *erefrrs7.inp*):

Displacements, curved beam solution

$u_x^B = 2.1735 \times 10^{-5}$, $u_y^B = 1.4570 \times 10^{-4}$.

Displacements, straight beam solution

$u_x^B = 1.6667 \times 10^{-5}$, $u_y^B = 0.0$.

Open section beam elements (see *erefois7.inp*):

Displacements, curved beam solution

$u_x^B = 3.1946 \times 10^{-4}$, $u_y^B = 1.0962 \times 10^{-3}$.

Displacements, straight beam solution

$u_x^B = 2.8153 \times 10^{-4}$, $u_y^B = 0.0$.

Pipe elements (see *erefpps7.inp*):

Displacements, curved beam solution

$u_x^B = 2.9461 \times 10^{-5}$, $u_y^B = 4.5078 \times 10^{-5}$.

Displacements, straight beam solution

$u_x^B = 2.7922 \times 10^{-5}$, $u_y^B = 0.0$.

Results and discussion

Table 1: Regular beams and shells.

Element Type	u_x^B	u_y^B	Remarks
B21 (1-element mesh)	1.6667×10^{-5}	0.0	Straight*
B21 (Refined mesh)	2.1715×10^{-5}	1.4343×10^{-4}	Curved
B21H (1-element mesh)	1.6667×10^{-5}	0.0	Straight*
B21H (Refined mesh)	2.1665×10^{-5}	1.4343×10^{-4}	Curved
B22	2.1085×10^{-5}	1.4686×10^{-4}	Curved
B22H	2.1085×10^{-5}	1.4686×10^{-4}	Curved
B23	2.0873×10^{-5}	1.4549×10^{-4}	Curved
B23H	2.0873×10^{-5}	1.4549×10^{-4}	Curved
B31 (1-element mesh)	1.6667×10^{-5}	0.0	Straight*
B31 (Refined mesh)	2.1715×10^{-5}	1.4343×10^{-4}	Curved
B31H (1-element mesh)	1.6667×10^{-5}	0.0	Straight
B31H (Refined mesh)	2.1715×10^{-5}	1.4343×10^{-4}	Curved
B32	2.1084×10^{-5}	1.4686×10^{-4}	Curved

Element Type	u_x^B	u_y^B	Remarks
B32H	2.1084×10^{-5}	1.4686×10^{-4}	Curved
B33	2.0873×10^{-5}	1.4548×10^{-4}	Curved
B33H	2.0873×10^{-5}	1.4548×10^{-4}	Curved
S4 (1-element mesh)	1.6292×10^{-5}	0.0	Straight*
S4 (Refined mesh)	2.1607×10^{-5}	1.4314×10^{-4}	Curved
S4R (1-element mesh)	1.6666×10^{-5}	0.0	Straight*
S4R (Refined mesh)	2.1661×10^{-5}	1.4340×10^{-4}	Curved
S4R5 (1-element mesh)	1.6666×10^{-5}	0.0	Straight*
S4R5 (Refined mesh)	2.1667×10^{-5}	1.4344×10^{-4}	Curved
S8R	2.1036×10^{-5}	1.4508×10^{-4}	Curved
S8R5	2.1001×10^{-5}	1.4638×10^{-4}	Curved
S9R5	2.1001×10^{-5}	1.4638×10^{-4}	Curved
STRI3	1.6667×10^{-5}	0.0	Straight
STRI65 †	2.0750×10^{-5}	1.4331×10^{-4}	Curved
SC6R	2.1673×10^{-5}	1.425×10^{-4}	Curved
SC8R	2.156×10^{-5}	1.2402×10^{-4}	Curved
SC8R	1.6608×10^{-5}	2.5028×10^{-4}	Straight
SC8R**	2.1491×10^{-5}	1.4175×10^{-4}	Curved
SC8R**	1.6276×10^{-5}	2.4271×10^{-4}	Straight
CSS8	2.1369×10^{-5}	1.4093×10^{-4}	Curved
CSS8	1.4511×10^{-5}	1.1925×10^{-4}	Straight

Table 2: Open section beam elements.

Element Type	u_x^B	u_y^B	Remarks
B31OS (1-element mesh)	2.8153×10^{-4}	0.0	Straight*
B31OS (Refined mesh)	3.2287×10^{-4}	1.0790×10^{-3}	Curved
B31OSH (1-element mesh)	2.8153×10^{-4}	0.0	Straight*
B31OSH (Refined mesh)	3.2287×10^{-4}	1.0790×10^{-3}	Curved
B32OS (1-element mesh)	3.1787×10^{-4}	1.1048×10^{-3}	Curved
B32OSH (Refined mesh)	3.1787×10^{-4}	1.1048×10^{-3}	Curved

Table 3: Pipe elements.

Element Type	u_x^B	u_y^B	Remarks
PIPE21 (1-element mesh)	2.7922×10^{-5}	0.0	Straight*
PIPE21 (Refined mesh)	2.9768×10^{-5}	4.4373×10^{-5}	Curved
PIPE21H (1-element mesh)	2.7922×10^{-5}	0.0	Straight*
PIPE21H (Refined mesh)	2.9768×10^{-5}	4.4373×10^{-5}	Curved

Element Type	u_x^B	u_y^B	Remarks
PIPE22 (1-element mesh)	2.9572×10^{-5}	4.5435×10^{-5}	Curved
PIPE22H (1-element mesh)	2.9572×10^{-5}	4.5435×10^{-5}	Curved
PIPE31 (1-element mesh)	2.7922×10^{-5}	0.0	Straight*
PIPE31 (Refined mesh)	2.9768×10^{-5}	4.4373×10^{-5}	Curved
PIPE31H (1-element mesh)	2.7922×10^{-5}	0.0	Straight*
PIPE31H (Refined mesh)	2.9768×10^{-5}	4.4373×10^{-5}	Curved
PIPE32 (1-element mesh)	2.9572×10^{-5}	4.5435×10^{-5}	Curved
PIPE32H (1-element mesh)	2.9572×10^{-5}	4.5435×10^{-5}	Curved

* These are first-order elements and are unable to capture initial curvature with a one-element mesh. However, a refined mesh for these elements yields very good results.† Due to the lack of symmetry for triangular meshes, the displacements at the nodes that are at point *B* may differ slightly. The maximum values are documented here.** These results are obtained using enhanced hourglass control.

Input files

Coarse mesh tests:

eb22rms7.inp	B21 elements.
eb2hrms7.inp	B21H elements.
eb23rms7.inp	B22 elements.
eb2irms7.inp	B22H elements.
eb2arms7.inp	B23 elements.
eb2jrms7.inp	B23H elements.
eb32rms7.inp	B31 elements.
eb3hrms7.inp	B31H elements.
ebo2ims7.inp	B31OS elements.
ebohims7.inp	B31OSH elements.
eb33rms7.inp	B32 elements.
eb3irms7.inp	B32H elements.
ebo3ims7.inp	B32OS elements.
eboiims7.inp	B32OSH elements.
eb3arms7.inp	B33 elements.
eb3jrms7.inp	B33H elements.
ep22pms7.inp	PIPE21 elements.
inicurv_pipe2d_xpl.inp	PIPE21 elements in Abaqus/Explicit.
ep2hpms7.inp	PIPE21H elements.
ep23pms7.inp	PIPE22 elements.
ep2ipms7.inp	PIPE22H elements.
ep32pms7.inp	PIPE31 elements.
inicurv_pipe3d_xpl.inp	PIPE31 elements in Abaqus/Explicit.
ep3hpms7.inp	PIPE31H elements.
ep33pms7.inp	PIPE32 elements.

ep3ipms7.inp	PIPE32H elements.
ese4sms7.inp	S4 elements.
esf4sms7.inp	S4R elements.
es54sms7.inp	S4R5 elements.
es68sms7.inp	S8R elements.
es58sms7.inp	S8R5 elements.
es59sms7.inp	S9R5 elements.
es63sms7.inp	STRI3 elements.
es56sms7.inp	STRI65 elements.
esc8sms7.inp	SC8R elements.
esc8sms7_eh.inp	SC8R elements with enhanced hourglass control.
ecss8sms7.inp	CSS8 elements.

Fine mesh tests:

eb22rfs7.inp	B21 elements.
eb2hrfs7.inp	B21H elements.
eb32rfs7.inp	B31 elements.
eb3hrfs7.inp	B31H elements.
ebo2ifs7.inp	B31OS elements.
ebohifs7.inp	B31OSH elements.
ep22pfs7.inp	PIPE21 elements.
ep2hpfs7.inp	PIPE21H elements.
ep32pfs7.inp	PIPE31 elements.
ep3hpfs7.inp	PIPE31H elements.
ese4sfs7.inp	S4 elements.
esf4sfs7.inp	S4R elements.
es54sfs7.inp	S4R5 elements.
esc6sfs7.inp	SC6R elements.
esc8sfs7.inp	SC8R elements.
esc8sfs7_eh.inp	SC8R elements with enhanced hourglass control.
ecss8sfs7.inp	CSS8 elements.

Normal definitions of beams and shells

Products: Abaqus/Standard Abaqus/Explicit

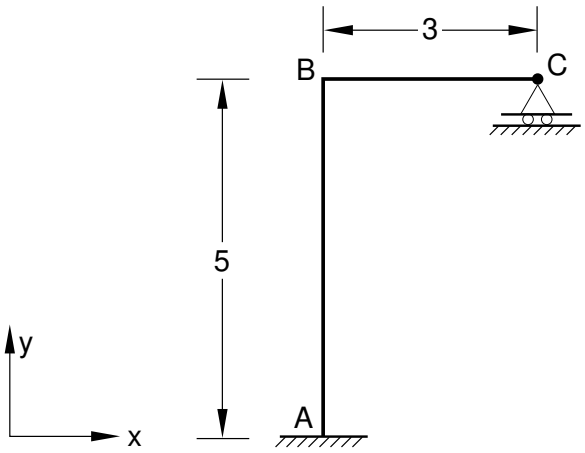
Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

B21 B21H B22 B22H B23 B23H B31 B31H B31OS B31OSH B32 B32H B32OS B32OSH B33 B33H
PIPE21 PIPE21H PIPE22 PIPE22H PIPE31 PIPE31H PIPE32 PIPE32H
S4 S4R S4R5 S8R S8R5 S9R5 STRI3 STRI65

Problem description



Material: Linear elastic, Young's modulus = 30×10^6.

Boundary conditions: End *A* is clamped, $u_y = 0$ at end *C*.

Loading: $F_x =$ 25.0 at end *C*.

Section properties: $A =$ 0.25, $I_{12} =$ 0, $I_{11} = I_{22} =$ 1.0×10^6, $J =$ 0.01041667.

For B23, B23H, B33, B33H, and STRI3 elements, different values of I_{22} are used: vertical members, $I_{22} = 5.208 \times 10^{-3}$, horizontal members, $I_{22} = 1.0 \times 10^{-6}$.

For pipe elements a circular cross-section of outer radius 0.5 and wall thickness 0.05 is used. Five pipe elements are used along segment AB. A single linear beam element is used along BC with section properties as defined above. In Abaqus/Explicit the loading is applied using a smooth step amplitude to achieve a nearly static response at steady state, similar to that in Abaqus/Standard.

Remarks

Normal definitions written to the output file by the analysis input file processor are all correct.

Reference solution

Displacements: $u_x^B = \left(Pl^3\right) / (3EI) + (Pl) / (kGA)$.

For shear flexible elements properties have been defined such that the first term is negligible.

For Love-Kirchhoff (cubic) elements the second term does not apply.

Results and discussion

Element Type	u_x^B (Abaqus)	u_x^B (Analytical)
B21(H)	5.098×10^{-5}	5.098×10^{-5}
B22(H)	5.098×10^{-5}	5.098×10^{-5}
B23(H)	6.662×10^{-3}	6.667×10^{-3}
B31(H)	5.098×10^{-5}	5.098×10^{-5}
B31OS(H)	5.098×10^{-5}	5.098×10^{-5}
B32(H)	5.098×10^{-5}	5.098×10^{-5}
B32OS(H)	5.098×10^{-5}	5.098×10^{-5}
B33(H)	6.662×10^{-3}	6.667×10^{-3}
PIPE21(H)	2.166 × 10–3	2.199 × 10–3
PIPE22(H)	2.167 × 10–3	2.199 ×10–3
PIPE31(H)	2.166×10^{-3}	2.199×10^{-3}
PIPE32(H)	2.167×10^{-3}	2.199×10^{-3}
S4	5.098×10^{-5}	5.098×10^{-5}
S4R	5.098×10^{-5}	5.098×10^{-5}
S4R5	5.098×10^{-5}	5.098×10^{-5}
S8R	5.098×10^{-5}	5.098×10^{-5}
S8R5	5.098×10^{-5}	5.098×10^{-5}
S9R5	5.098 × 10–5	5.098 × 10–5
STRI3 †	6.341×10^{-3}	6.667×10^{-3}
STRI65 †	3.991×10^{-5}	5.098×10^{-5}

† Due to the lack of symmetry for triangular meshes, the displacements at the nodes that are at point *B* differ slightly. The maximum values are documented here. For pipe elements in Abaqus/Explicit the results are very close to those obtained with Abaqus/Standard; the small differences can be attributed to steady-state oscillations.

Input files

eb22gxs8.inp B21 elements.

eb22gxs8_normal.inp	B21 elements with user-specified nodal normals.
eb2hgxs8.inp	B21H elements.
eb23gxs8.inp	B22 elements.
eb2igxs8.inp	B22H elements.
eb2agxs8.inp	B23 elements.
eb2jgxs8.inp	B23H elements.
eb32gxs8.inp	B31 elements.
eb32gxs8_normal.inp	B31 elements with user-specified nodal normals.
eb3hgxs8.inp	B31H elements.
ebo2gxs8.inp	B31OS elements.
ebohgxs8.inp	B31OSH elements.
eb33gxs8.inp	B32 elements.
eb3igxs8.inp	B32H elements.
ebo3gxs8.inp	B32OS elements.
eboigxs8.inp	B32OSH elements.
eboigxs8_normal.inp	B32OSH elements with user-specified fixed nodal normals.
eb3agxs8.inp	B33 elements.
eb3jgxs8.inp	B33H elements.
eb3jgxs8_normal.inp	B33H elements with user-specified nodal normals.
ep22pxs8.inp	PIPE21 elements.
ep2hpxs8.inp	PIPE21H elements.
ep23pxs8.inp	PIPE22 elements.
ep2ipxs8.inp	PIPE22H elements.
ep32pxs8.inp	PIPE31 elements.
ep3hpxs8.inp	PIPE31H elements.
ep33pxs8.inp	PIPE32 elements.
ep3ipxs8.inp	PIPE32H elements.
ese4sgs8.inp	S4 elements.
esf4sgs8.inp	S4R elements.
es54sgs8.inp	S4R5 elements.
es68sgs8.inp	S8R elements.
es58sgs8.inp	S8R5 elements.
es59sgs8.inp	S9R5 elements.
es63sgs8.inp	STRI3 elements.
es56sgs8.inp	STRI65 elements.
normdef_pipe2d_xpl.inp	PIPE21 elements in Abaqus/Explicit.
normdef_pipe3d_xpl.inp	PIPE31 elements in Abaqus/Explicit.

Constant curvature test for shells

Products: Abaqus/Standard

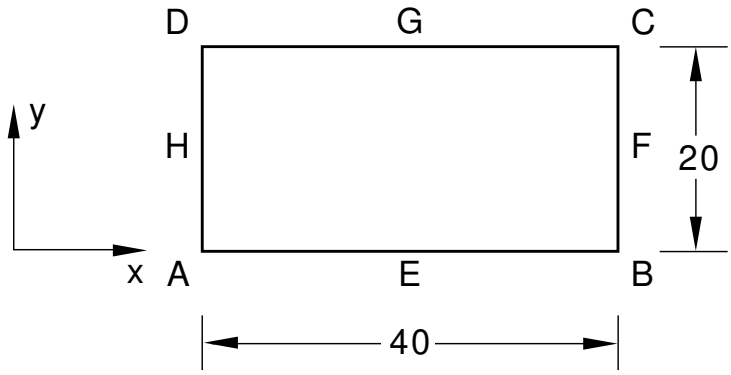
Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

S3 S3R S4 S4R S4R5 S8R S8R5 S9R5 STRI3 STRI65

Problem description



Material: Linear elastic, Young's modulus = 1×10^3, Poisson's ratio = 0.3.

Boundary conditions: $u_z = 0$ at nodes *A*, *B*, and *D*, $u_x = u_y = 0$ at all nodes along the perimeter.

Loading: $F_z = -2.0$ at node *C*, $M_x = 20.0$ at nodes *A* and *B*, $M_x = -20.0$ at nodes *C* and *D*, $M_y = 10.0$ at nodes *B* and *C*, $M_y = -10.0$ at nodes *A* and *D*.

Reference solution

Displacements: $u_z^C = -12.48$.

Results and discussion

Element type	u_z^C
S3/S3R	−12.51
S4R	−12.54
S4	−12.54

Element type	u_z^C
S4R5	−12.496
S8R*	−12.555
S8R5	−12.527
S9R5	−12.527
STRI3	−12.480
STRI65	−12.545

*A refined mesh consisting of two elements is used for the S8R model since hourglassing occurs in a one-element mesh.

Input files

esf3sxs9.inp	S3/S3R elements.
ese4sxs9.inp	S4 elements.
esf4sxs9.inp	S4R elements.
es54sxs9.inp	S4R5 elements.
es68sxs9.inp	S8R elements.
es58sxs9.inp	S8R5 elements.
es59sxs9.inp	S9R5 elements.
es63sxs9.inp	STRI3 elements.
es56sxs9.inp	STRI65 elements.

Verification of section forces for shells

Products: Abaqus/Standard

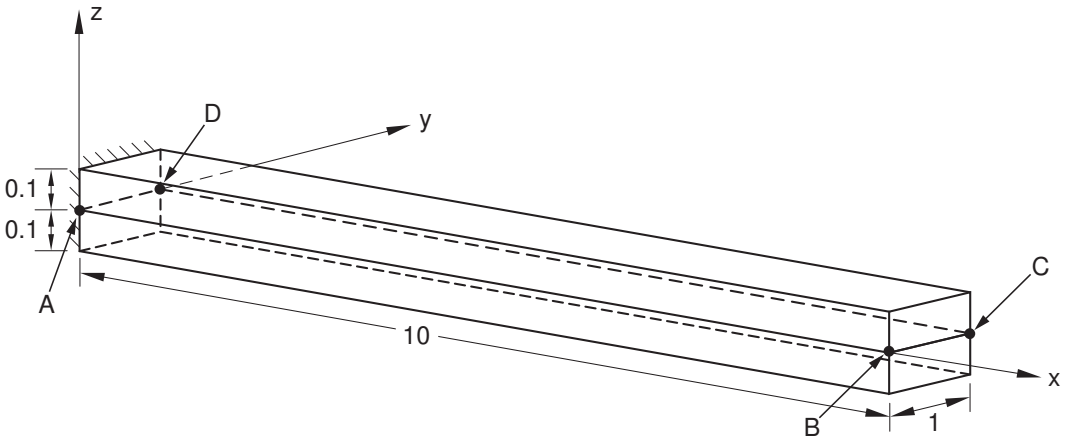
Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

S4 S4R S4R5 S8R S8R5 S9R5 STRI3 STRI65

Problem description



Material: Linear elastic, $E_{11} = 2.00313 \times 10^7$, $E_{22} = 5.00783 \times 10^5$, $E_{12} = 1.25296 \times 10^5$, $G_{13} = 0.5 \times 10^5$, $G_{23} = 0.5 \times 10^5$.

Boundary conditions: Nodes along edge AD are clamped.

Loading: $F_z = 0.5$ at nodes *B* and *C*.

Orientations: 90° in the first layer and 0° in the second layer, with respect to the *x*-axis, rotated about the *z*-axis.

There are two elements with identical geometries in the model. The first element is defined using a composite shell section and uses a local coordinate system. The second element is defined using a general shell section, with the section stiffness matrix input directly, and is equivalent to the two-layer model presented above.

The section stiffness is:

$$\mathbf{D} = \begin{bmatrix} 0.2053 \times 10^7 & 0.2506 \times 10^5 & 0.0 & 0.9765 \times 10^5 & 0.0 & 0.0 \\ & 0.2053 \times 10^7 & 0.0 & 0.0 & -0.9765 \times 10^5 & 0.0 \\ \text{Symmetric} & & 0.5 \times 10^5 & 0.0 & 0.0 & 0.0 \\ & & & 0.6844 \times 10^4 & 0.8346 \times 10^2 & 0.0 \\ & & & & 0.6844 \times 10^4 & 0.0 \\ & & & & & 0.1667 \times 10^3 \end{bmatrix}$$

Reference solution

Stress resultants: Moment = $-1.0(10.0 - x)$.

Results and discussion

All elements yield acceptable solutions. Local coordinate directions are requested in the input file with element type S8R5 (*es58s2sc.inp*).

Input files

ese4s2sc.inp	S4 elements.
esf4s2sc.inp	S4R elements.
es54s2sc.inp	S4R5 elements.
es68s2sc.inp	S8R elements.
es58s2sc.inp	S8R5 elements.
es59s2sc.inp	S9R5 elements.
es63s2sc.inp	STRI3 elements.
es56s2sc.inp	STRI65 elements.

Composite shell sections

Products: Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

S4 S4R S4RS S4RSW

Features tested

Shell general section, composite laminate, orientation.

Problem description

There are three different options for defining a composite shell section within Abaqus/Explicit:

1. A shell general section in which the user supplies the (constant) stiffness coefficients for the shell section in matrix form.
2. A layered, elastic shell section, for which Abaqus/Explicit calculates a pre-integrated effective shell stiffness matrix. With this option the user defines the number of layers, the material properties for each layer, and the orientation in each layer. The material definition must be elastic to pre-integrate the shell stiffnesses. This option will print the matrix of effective stiffness coefficients that are calculated from the layered shell section.
3. A numerically integrated shell section. The shell section definition for this case is basically the same as for option (b) above: the user defines the number of layers, the material properties for each layer, the orientation in each layer, and the number of integration points through the thickness of each layer. The material properties for this case may be nonlinear (e.g., plasticity may be used). If only elastic properties are used with a shell section, it is more efficient to use a general shell section as in option (b) above.

The purpose of this verification problem is to ensure that each of the different options for generating a shell section gives the same results for the same physical shell model. The test consists of six identical simply supported beams under uniform pressure loading. Two sets of analyses are performed: one in which the beams are modeled with S4R elements and the other in which the beams are modeled with S4RS elements. Due to symmetry only one-half of each beam is considered. Six cases are studied for each element type:

1. A sandwich beam modeled with the numerically integrated shell section. There are three linear elastic layers consisting of an aluminum layer (thickness 8 mm) sandwiched between two steel layers (thickness 6 mm). Each layer has three material points through the thickness.
2. The same sandwich beam as Case 1, modeled with a composite general shell section.
3. The same sandwich beam as Case 1, modeled with a general shell section, where the stiffness matrix (21 coefficients) of the shell section is given with values corresponding to the pre-integrated Case 2.
4. The same as Case 1 except that an in-plane orientation angle of 90° is applied to each layer. Since the material is isotropic, the orientation should not affect the final results.
5. The same as Case 2 except that an in-plane orientation angle of 90° is applied to each layer.
6. The same as Case 3 except that an orientation is applied to the whole section. The in-plane orientation is defined with the local node numbers.

Results and discussion

Figure 1 shows the contour plots of section moment SM1 on the deformed geometry for Cases 1 through 5 and section moment SM2 for Case 6 when the analysis is performed using the S4R element. *Figure 2* shows the histories of the central deflection of the beam for all six cases. *Figure 3* shows the histories of the section force SF1 (membrane force) at the center of the beams. Note that in Abaqus/Explicit any orientation option will not affect the output of section forces as they will always be in the default shell system. The stresses and strains are output to the selected results file in the local material coordinate system. The directions of the local coordinate system for these quantities are automatically written to the results file.

Figure 4 through *Figure 6* show the analogous results for the analysis performed using S4RS elements.

Input files

shellsect.inp	S4R model with a shell section.
shellgensect.inp	S4R model with a general shell section.
shellsect_s4rs.inp	S4RS model with a shell section.
shellgensect_s4rs.inp	S4RS model with a general shell section.
shellsect_s4.inp	S4 model with a shell section included for the purpose of testing performance only.
shellgensect_s4.inp	S4 model with a general shell section included for the purpose of testing performance only.
shellgensect_s4rsw.inp	S4RSW model with a general shell section included for the purpose of testing performance only.

Figures

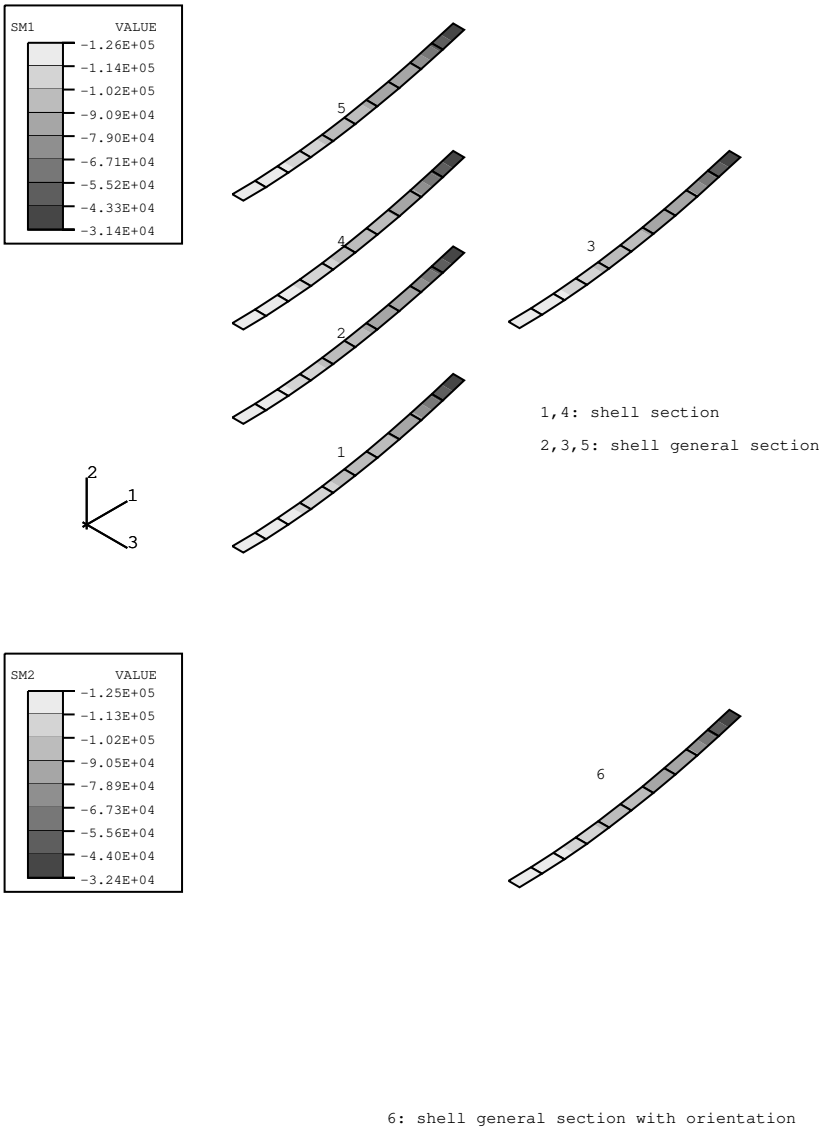


Figure 1: Contours of bending moment at 12 milliseconds (S4R model).

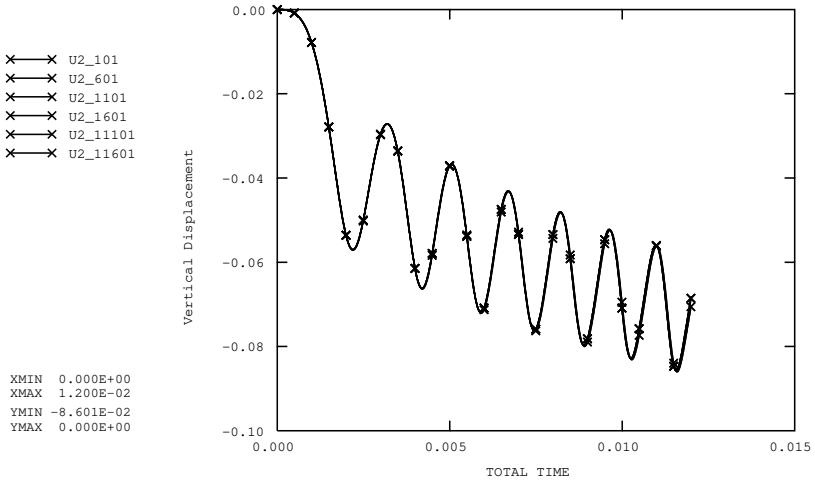


Figure 2: Histories of the central deflections (S4R model).

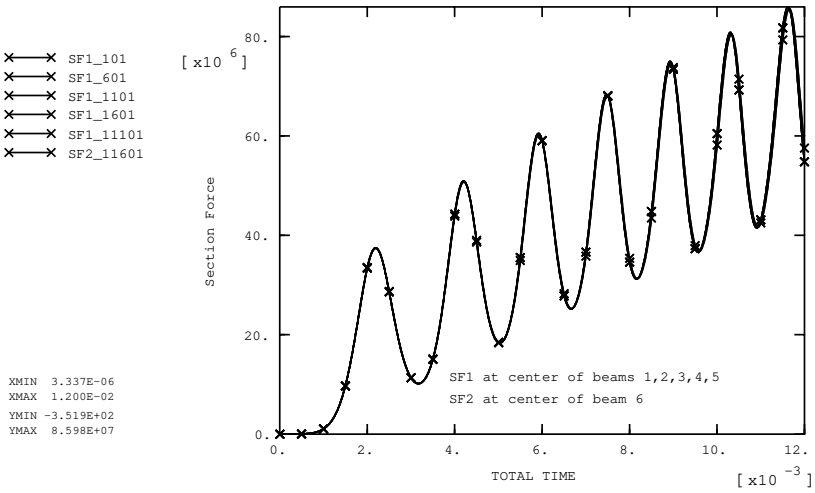


Figure 3: Histories of the membrane forces (S4R model).

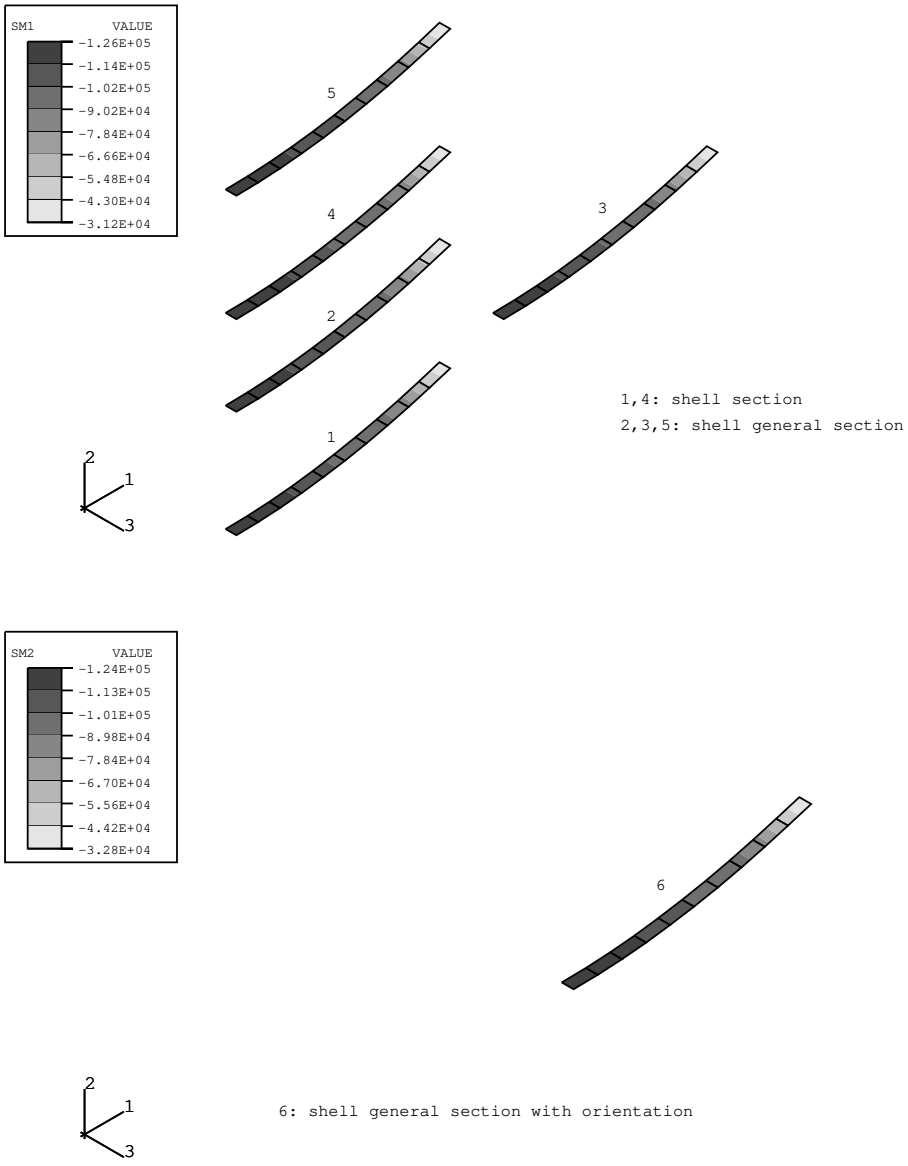


Figure 4: Contours of bending moment at 12 milliseconds (S4RS model).

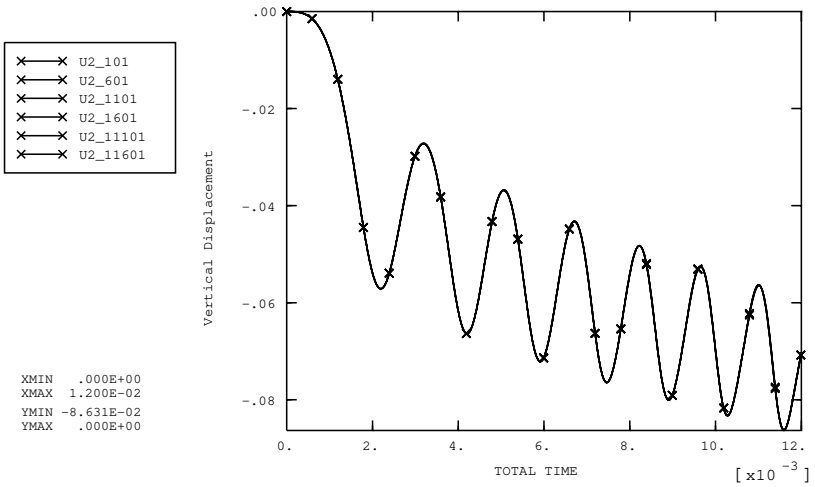


Figure 5: Histories of the central deflections (S4RS model).

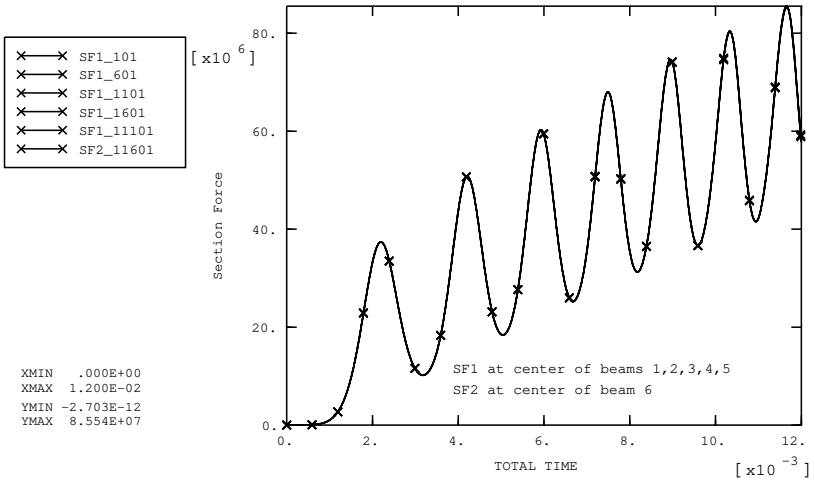


Figure 6: Histories of the membrane forces (S4RS model).

Cantilever sandwich beam: shear flexible shells

Products: Abaqus/Standard

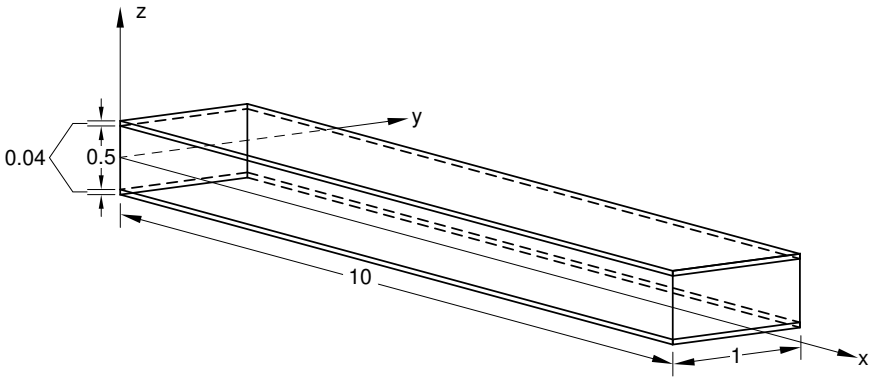
Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

S4 S4R S8R S4T S4RT S8RT

Problem description



Material: For the face a linear elastic material with Young's modulus = 1.0×10^7 and Poisson's ratio = 0 is modeled. For the core the transverse shear moduli are given as 1.0×10^4 and all other properties in the plane are set to negligible values, using the LAMINA definition.

Boundary conditions: All nodes are clamped at one end.

Loading: $F_z =$ 750.0 distributed consistently to the nodes at the free end.

General: Gauss integration is used for the shell cross-section for the S4, S4R, and S8R elements.

Simpson integration is used for the shell cross-section for the S4T, S4T, S4RT, and S8RT elements.

Reference solution

Displacement at the free end (Plantema, *Sandwich Construction*, John Wiley and Sons, Inc., 1966): u_z= 5.5684.

Maximum bending stress at the top of the clamped end, for the case of warping prevention as enforced here: σ_{11} = 3.7275×10^5.

Results and discussion

Element Type	u_z	σ_{11}
S4	5.55	3.5136×10^5
S4R	5.55	3.5136×10^5
S8R	5.56	3.6439×10^5
S4T	5.55	3.537×10^5
S4RT	5.55	3.537×10^5
S8RT	5.56	3.6439×10^5

Input files

ese4scsi.inp	S4 elements.
esf4scsi.inp	S4R elements.
es68scsi.inp	S8R elements.
es34tcsi.inp	S4T elements.
es4rtcsi.inp	S4RT elements.
es38tcsi.inp	S8RT elements.

Thermal stress in a cylindrical shell

Products: Abaqus/Standard Abaqus/Explicit

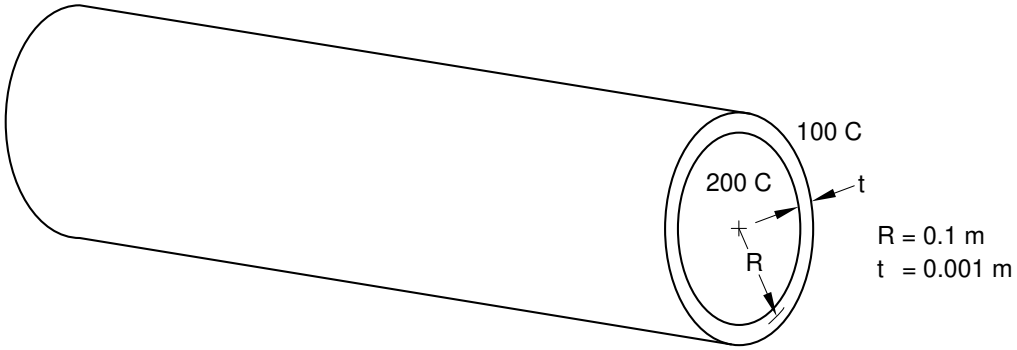
Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

DSAX1 DSAX2 DS3 DS4 DS6 DS8 DCAX8 DC3D20
SAX1 SAX2 SAX2T STRI65 S4R5 S8R5
S4RT S8RT
CAX3T CAX4RT CAX4RHT CAX8R CAX8RT
CGAX4RT CGAX8RT CGAX4RHT
C3D4T C3D6T C3D8T C3D8RT C3D20R C3D20RT

Problem description



The cylindrical shell is shown above. A single element is used in the Abaqus/Standard analyses and in the Abaqus/Explicit analysis using the coupled thermal shell element. In the Abaqus/Explicit analyses that use solid elements, two elements are used in the radial direction. For the nonaxisymmetric elements the element subtends an angle of 11.25° at the center, which is equivalent to 32 elements around the circumference.

Steady-state conditions are assumed in the Abaqus/Standard simulation. A transient simulation is performed in Abaqus/Explicit. The total simulation time is 0.4 seconds for the analyses using solid elements, and 0.06 seconds for the analysis using a shell element. This provides enough time for the transient solution to reach steady-state conditions in this problem. Mass scaling is used for the solid element analyses to reduce the computational cost of the Abaqus/Explicit analyses.

Material:

Density	7800 kg/m^3
Conductivity	52 J/ms °C
Specific heat	586 J/kg °C
Thermal expansion coefficient	1.2×10^{-5}
Young's modulus	200×10^3 MPa

Poisson's ratio	0.3

Boundary conditions: For the thermal analyses the temperatures of the inside and outside surfaces are prescribed to be 200°C and 100°C, respectively. For the stress analyses the rotation vector in the circumferential direction is constrained, but the cylinder is free to expand axially. For the continuum element meshes equations are used to provide the rotational constraints. For the nonaxisymmetric cases symmetrical constraints are applied in the circumferential direction to model the complete cylinder.

In the Abaqus/Explicit simulations the temperatures are applied gradually to ensure a quasi-static response.

General: For all of the analyses except those using the coupled temperature-displacement elements (SAX2T, S8RT, CAX4RT, CAX4RHT, CGAX4RT, CGAX4RHT, CAX8RT, CGAX8RT, and C3D20RT in Abaqus/Standard and S4RT, CAX3T, CAX4RT, C3D4T, C3D6T, C3D8RT, and C3D8T in Abaqus/Explicit), the analyses are run in pairs: a thermal analysis followed by its corresponding stress analysis.

Gauss integration is used for the shell cross-section for input file *es54sxsj.inp*.

Reference solution

The temperature distribution through the thickness of the cylinder is given by

$$T = T_i + \frac{(T_0 - T_i)}{\ln(R_0/R_i)} \ln(R/R_i),$$

where R_0 is the outer radius, R_i is the inner radius, T_0 is the outside temperature, and T_i is the inside temperature.

The analytical solution for the stresses is given in Chapter 15 of "Theory of Plates and Shells," second edition, by Timoshenko and Woinowsky-Krieger. The stresses at the outer and inner surfaces are given by

$$\sigma = \pm \frac{E\alpha(T_i - T_0)}{2(1-\nu)},$$

where E is Young's modulus, α is the coefficient of thermal expansion, and ν is Poisson's ratio. The upper sign refers to the outer surface, indicating that a tensile stress will act on this surface if $T_i > T_0$.

This gives a theoretical stress of 171.43 MPa.

Results and discussion

The axisymmetric and second-order shell elements agree exactly with the theory. The first-order three-dimensional shells (S4R5) show an error of −5.1%. The continuum elements show small discrepancies (< 1%) from the reference solution.

The results obtained with Abaqus/Explicit are in close agreement with the analytical solution and with those obtained with Abaqus/Standard.

Input files

Abaqus/Standard input files

esa2dxsj.inp	DSAX1 elements.
esa3dxsj.inp	DSAX2 elements.
es33dxsj.inp	DS3 elements.

es34dxsj.inp	DS4 elements.
es36dxsj.inp	DS6 elements.
es38dxsj.inp	DS8 elements.
eca8dfsj.inp	DCAX8 elements.
ec3kdfsj.inp	DC3D20 elements.
esa2sxsj.inp	SAX1 elements.
esa3sxsj.inp	SAX2 elements.
es56sxsj.inp	STRI65 elements.
es54sxsj.inp	S4R5 elements.
es58sxsj.inp	S8R5 elements.
eca8srsj.inp	CAX8R elements.
ec3ksrsj.inp	C3D20R elements.
esa3txsj.inp	SAX2T elements.
es34txsj.inp	S4T elements.
es4rtxsj.inp	S4RT elements.
es38txsj.inp	S8RT elements.
ecax3tsj.inp	CAX3T elements.
eca4trsj.inp	CAX4RT elements.
eca4tysj.inp	CAX4RHT elements.
eca4hrsj.inp	CGAX4RT elements.
eca4hysj.inp	CGAX4RHT elements.
eca8trsj.inp	CAX8RT elements.
eca8hrsj.inp	CGAX8RT elements.
ec3ktrsj.inp	C3D20RT elements.
thermstresscyl_std_c3d4t.inp	C3D4T elements.
thermstresscyl_std_c3d6t.inp	C3D6T elements.

Abaqus/Explicit input files

thermstresscyl_xpl_cax3t.inp	CAX3T elements.
thermstresscyl_xpl_cax4rt.inp	CAX4RT elements.
thermstresscyl_xpl_c3d4t.inp	C3D4T elements.
thermstresscyl_xpl_c3d6t.inp	C3D6T elements.
thermstresscyl_xpl_c3d8rt.inp	C3D8RT elements.
thermstresscyl_xpl_c3d8t.inp	C3D8T elements.
thermstresscyl_xpl_s4rt.inp	S4RT elements.

Variable thickness shells and membranes

Products: Abaqus/Standard

Overview

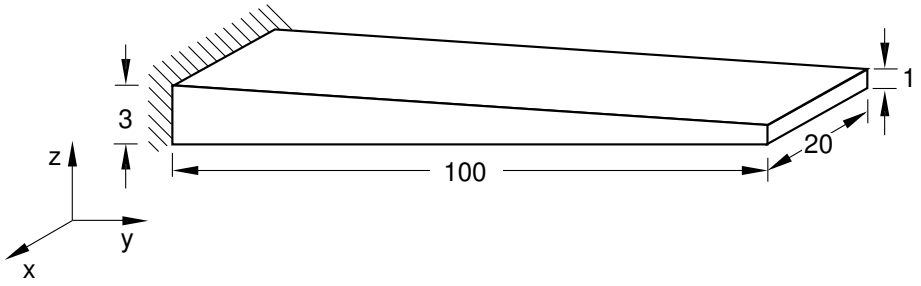
This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

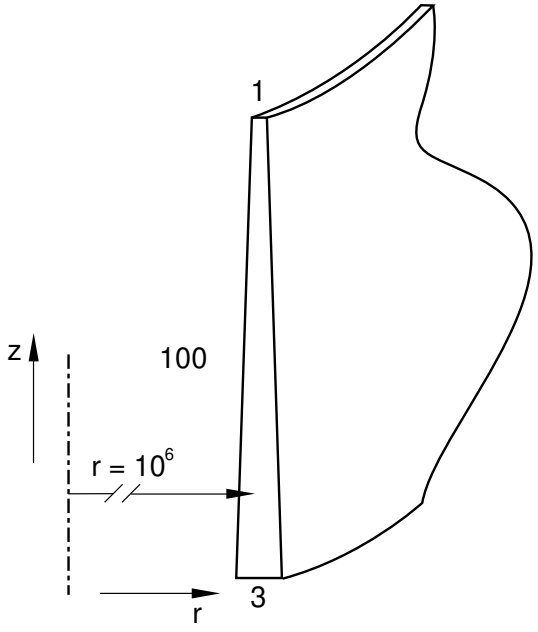
S4 S4R S4R5 S8R S8R5 S9R5 STRI3 STRI65
SAX1 SAX2 SAXA1*n* SAXA2*n*
S4T S4RT S8RT SAX2T
DS3 DS4 DS6 DS8 DSAX1 DSAX2
M3D3 M3D4 M3D4R M3D6 M3D8 M3D8R M3D9 M3D9R
MAX1 MAX2 MGAX1 MGAX2 MCL6 MCL9

Problem description

For the three-dimensional shell and membrane elements (except the cylindrical membrane elements), the model consists of a tapered plate of length 100 and width 20. The plate is clamped at one end, and the thickness varies linearly across the plate from 3 at the clamped end to 1 at the free end. The first-order models consist of 10 elements along the length and two across the width; the second-order models consist of five elements along the length and one across the width.



For the axisymmetric elements and the cylindrical membrane elements, the model consists of a tapered cylinder with a radius of 1×10^6 and a length of 100. The cylinder is clamped at one end, and the thickness varies linearly along the length of the cylinder from 3 at the clamped end to 1 at the free end. The radius is chosen to be very large to ensure that the effects of circumferential stresses are negligible. The cylinder is meshed with ten first-order elements or five second-order elements.



Material: For stress analysis: linear elastic, Young's modulus = 1000, Poisson's ratio = 0; for heat transfer: conductivity = 1.

Boundary conditions: Clamped at the end with thickness 3.

Loading:

Shell bending model

Bending moment of 3 per unit length at the thin end of the shell.

Membrane tension model

In-plane force of 50 per unit length at the thin end of the membrane.

Axisymmetric membrane tension model

In-plane force of 50 per unit length at the thin end of the membrane.

Cylindrical membrane tension model

In-plane force of 50 per unit length at the thin end of the membrane. A beam type multi-point constraint is used to tie all nodes at the thin end of the membrane to a main node. The load is then applied to the main node. This problem is set up using the symmetric model generation capability, with the corresponding axisymmetric problem as the base model.

Heat transfer shell model

Prescribed temperature = 0 at the thick end, prescribed temperature = 100 at the thin end.

Reference solution

Shell bending model

Tip displacement $u_z = 20.0$, tip rotation $\phi_x = 0.8$.

Membrane tension model

Tip displacement $u_y = 2.7465$.

Axisymmetric membrane tension model

Tip displacement $u_2 = 2.7465$.

Cylindrical membrane tension model

Tip displacement $u_3 = 2.7465$.

Heat transfer shell model

Temperatures $\theta(20) = 13.03$, $\theta(40) = 28.23$, $\theta(60) = 46.50$, $\theta(80) = 69.37$.

Results and discussion

All numerical solutions agree closely with the analytical solutions. The maximum error is about 1%. Local coordinate directions are used in input files *es34dnsq.inp* and *em34sfsq.inp*.

Input files

esf3snsq.inp	S3R elements.
ese4snsq.inp	S4 elements.
esf4snsq.inp	S4R elements.
es54snsq.inp	S4R5 elements.
es68snsq.inp	S8R elements.
es58snsq.inp	S8R5 elements.
es59snsq.inp	S9R5 elements.
es63snsq.inp	STRI3 elements.
es56snsq.inp	STRI65 elements.
esa2snsq.inp	SAX1 elements.
esa3snsq.inp	SAX2 elements.
esnssnsq.inp	SAXA11 elements.
esntsnsq.inp	SAXA12 elements.
esnusnsq.inp	SAXA13 elements.
esnvsnsq.inp	SAXA14 elements.
esnwsnsq.inp	SAXA21 elements.
esnxsnsq.inp	SAXA22 elements.

esnysnsq.inp	SAXA23 elements.
esnzsnsq.inp	SAXA24 elements.
es34tnsq.inp	S4T elements.
es4rtnsq.inp	S4RT elements.
es68tnsq.inp	S8RT elements.
esa3tnsq.inp	SAX2T elements.
es33dnsq.inp	DS3 elements.
es34dnsq.inp	DS4 elements.
es36dnsq.inp	DS6 elements.
es38dnsq.inp	DS8 elements.
esa2dnsq.inp	DSAX1 elements.
esa3dnsq.inp	DSAX2 elements.
em33sfsq.inp	M3D3 elements.
em34sfsq.inp	M3D4 elements.
em34srsq.inp	M3D4R elements.
em36sfsq.inp	M3D6 elements.
em38sfsq.inp	M3D8 elements.
em38srsq.inp	M3D8R elements.
em39sfsq.inp	M3D9 elements.
em39srsq.inp	M3D9R elements.
ema2srsq.inp	MAX1 elements.
ema3srsq.inp	MAX2 elements.
emg2srsq.inp	MGAX1 elements.
emg3srsq.inp	MGAX2 elements.
emc6srsq.inp	MCL6 elements.
emc9srsq.inp	MCL9 elements.

Shell offset

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

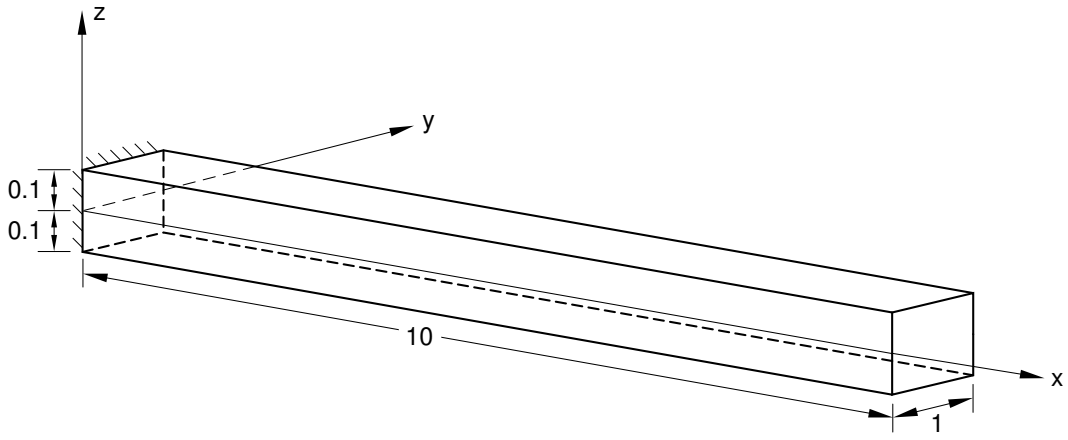
Elements tested

S4R S8R S8RT

Features tested

Shell offset used with a shell cross-section and a shell general section.

Problem description



The model consists of a plate with a length of 10.0, a width of 1.0, and a thickness of 0.2. The end at $x = 0$ is fixed, and all degrees of freedom except the rotation about the y-axis are constrained at $x = 10$. A rotation $\phi_y = -0.1$ is applied at $x = 10$ for the static analyses. A single shell element with an offset of half the shell's thickness from the midsurface is used to model the plate. Simpson's rule is used for the shell cross-section for all the elements.

Two additional input files (*esf4sxsd.inp* and *esf4sgsb.inp*) test the bending of a cantilevered half-cylinder. The model has a radius of 5, a length of 20, and a thickness of 0.2. One end is completely constrained, and a uniform upward pressure is applied to all the elements. A general, nonlinear static procedure is included.

Elastic material properties are used to define a material with $E = 3.0 \times 10^6$ and $\nu = 0.25$ in all cases.

Results and discussion

The verification of the shell offset results is based on the formulation described in *Transverse shear stiffness in composite shells and offsets from the midsurface*. The results are verified by comparing them to the results

obtained from an equivalent model without offset. This equivalent model is defined using a composite shell section and a general shell section, where an extra layer that has a negligible material modulus is added to the model.

Input files

Composite shell section offset:

esf4sxsc.inp	S4R elements; static steps.
esf4sxsd.inp	S4R elements; static steps using NLGEOM.
es68sxsc.inp	S8R elements; static steps.
es68sxsd.inp	S8R elements; frequency, steady-state dynamics, modal dynamic, and response spectrum steps.
es68sxra.inp	S8R elements; static steps with rebars.
es68sxrb.inp	S8R elements; frequency, steady-state dynamics, modal dynamic, and response spectrum steps with rebars.
es68sxxa.inp	S8R elements; static steps with thermal expansion.
es68sxxb.inp	S8RT elements; coupled temperature-displacement steps with static loading.

General shell section offset:

esf4sgsa.inp	S4R elements; static steps.
esf4sgsb.inp	S4R elements; static steps using NLGEOM.
es68sgsa.inp	S8R elements; static steps.
es68sgsb.inp	S8R elements; frequency, steady-state dynamics, modal dynamic, and response spectrum steps.

Axisymmetric membrane elements

Products: Abaqus/Standard

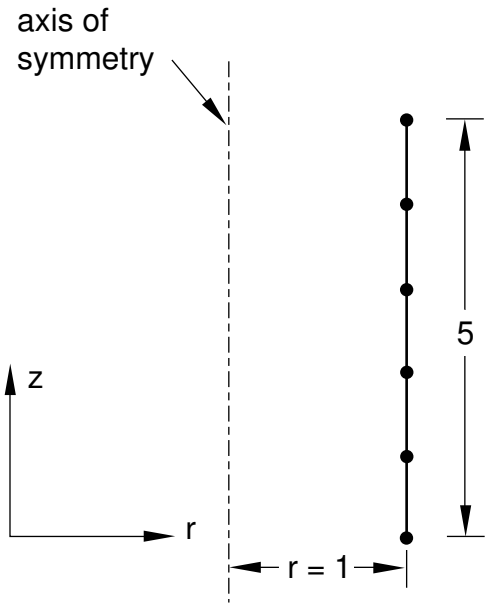
Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

MAX1 MAX2 MGAX1 MGAX2

Problem description



Model: Thickness of membrane is 0.05.

Material: For tests without orientation: linear elastic, Young's modulus = 10^5, Poisson's ratio = 0.3, thermal expansion coefficient = 10^{-7}.

For tests using orientation: linear elastic, engineering constants with $E_1 = 10^2$, $E_2 = 10^8$, $E_3 = 10^2$, $\nu_{12} = \nu_{13} = \nu_{23} = 0$, and $G_{12} = G_{13} = G_{23} = 10^2$.

Boundary conditions: Degree of freedom 2 is fixed for the bottom node. In addition, degree of freedom 5 is fixed for the bottom node for elements supporting twist.

Initial conditions: For tests without orientation an initial stress field of $\sigma_{11} = 0.001$ and $\sigma_{22} = 0.001$ is applied to all elements. The temperature of all nodes is set to 0 initially.

History definition I (for all element types)

Step 1 (geometrically nonlinear):

Loading: All degrees of freedom at all nodes are constrained. This step is recommended to apply the initial stresses. In subsequent steps all the necessary boundary conditions are applied.

Step 2 (perturbation):

Loading: A concentrated force (in direction 2) of magnitude 314 is applied to the top node.

Analytical solution: u_2 at top node = 0.04998.

Step 3 (perturbation):

Loading: Internal pressure of magnitude 500.

Analytical solution: Hoop stress = 10000.

Step 4 (perturbation):

Loading: The temperature at all nodes is increased to 5000.

Analytical solution: $\epsilon_{11} = \epsilon_{22} =$ 0.0005.

History definition II (for element types MGAX1 and MGAX2)

Step 1 (geometrically nonlinear):

Loading: All degrees of freedom at all nodes are constrained. This step is recommended to apply the initial stresses. In subsequent steps all the necessary boundary conditions are applied.

Step 2 (perturbation):

Loading: A concentrated moment (in degree of freedom 5) of magnitude 200 is applied to the top node.

Analytical solution: Shear stress = 636.22.

History definition III (for element types MGAX1 and MGAX2 using a local coordinate system)

Step 1 (geometrically nonlinear):

Loading: A concentrated load of magnitude 2 is applied to the top node.

Step 2 (geometrically nonlinear):

Loading: A concentrated moment of magnitude 2 is applied to the top node.

Results and discussion

History definition I:

All elements yield exact solutions.

History definition II:

The deviation from the analytical solution is approximately 0.06%.

History definition III:

The results are compared with those from a similar well-refined model using CGAX4R (axisymmetric continuum elements that support twist) elements. Since the strain in the thickness direction is very small in the continuum model, the section Poisson's ratio is set to 0 for the membrane model. The results match very well.

Input files

ema2srs3.inp	MAX1 elements.
ema3srs3.inp	MAX2 elements.
emg2srs3.inp	MGAX1 elements without twist.
emg3srs3.inp	MGAX2 elements without twist.
emg2srt3.inp	MGAX1 elements with twist.
emg3srt3.inp	MGAX2 elements with twist.
emg2sro3.inp	MGAX1 elements with **ORIENTATION*.
emg3sro3.inp	MGAX2 elements with **ORIENTATION*.

Cylindrical membrane elements

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

MCL6 MCL9

Problem description

Model: The model consists of a cylinder with initial radius and height both equal to 1. The initial thickness is 0.05. The cylinder is modeled using four cylindrical membrane elements, with each element spanning a 90° segment. Cylindrical transformation is used at all the nodes such that the boundary conditions and loads can be conveniently defined in the local radial, circumferential, and axial directions.

Material: For tests without orientation: linear elastic, Young's modulus = 10^5, Poisson's ratio = 0.3, thermal expansion coefficient = 10^{-7}.

For tests using orientation: linear elastic, engineering constraints with $E_1 = 10^2$, $E_2 = 10^8$, $E_3 = 10^2$, $\nu_{12} = \nu_{13} = \nu_{23} = 0$, and $G_{12} = G_{13} = G_{23} = 10^2$. The orientation is defined such that the fibers line up at an angle of 4° relative to the axial direction. With this setup, an axial force results in twist and, hence, development of shear strains.

Boundary conditions: The boundary conditions are different in the different steps and are described in the history definition subsection.

Initial conditions: For all the tests an initial stress field of $\sigma_{11} = 0.001$ and $\sigma_{22} = 0.001$ is applied to all elements. For tests that include thermal expansion the temperature of all nodes is set to 0 initially.

History definition 1 (for all element types)

Step 1 (geometrically nonlinear):

Loading: All degrees of freedom at all nodes are constrained. This step is recommended to apply the initial stresses. In subsequent steps all the necessary boundary conditions are applied.

Step 2 (geometrically nonlinear):

Boundary conditions: All nodes are fixed in the radial and circumferential directions. The top nodes are moved axially by 0.2.

Analytical solution: ϵ_{22} = 0.1823. The current membrane thickness is 0.04167.

Step 3 (geometrically nonlinear):

Boundary conditions: Same as in Step 2 except that the radial motion of all nodes is unconstrained.

Analytical solution: The axial strain remains unchanged. The radius and the thickness of the cylinder change in a manner such that the total volume is preserved.

History definition 2 (for all element types)

Step 1 (geometrically nonlinear):

Loading: All degrees of freedom at all nodes are constrained. This step is recommended to apply the initial stresses. In subsequent steps all the necessary boundary conditions are applied.

Step 2 (geometrically nonlinear):

Loading and boundary conditions: All nodes are fixed in the circumferential direction. In addition, all nodes at the bottom of the cylinder are fixed in the axial direction. The radial motion of all nodes are left unconstrained. Concentrated loads, which were obtained as reaction forces (at the bottom nodes of the cylinder) for the deformation state in history definition 1, are applied on the nodes on top of the cylinder.

Analytical solution: The deformation should be consistent with that at the end of Step 3 in history definition 1.

History definition 3 (for all element types)

Step 1 (geometrically nonlinear)

Loading: All degrees of freedom at all nodes are constrained. This step is recommended to apply the initial stresses. In subsequent steps all the necessary boundary conditions are applied.

Step 2 (perturbation):

Loading and boundary conditions: All nodes are fixed in the circumferential direction. In addition, all nodes at the bottom of the cylinder are fixed in the axial direction. The radial motion of all nodes is left unconstrained. An axial displacement of magnitude 0.2 is applied to all the nodes on the top of the cylinder.

Analytical solution: $\epsilon_{22} = 0.2$.

Step 3 (perturbation):

Loading and boundary conditions: All nodes are fixed in the circumferential direction. In addition, all nodes at the bottom of the cylinder are fixed in the axial direction. The radial motion of all nodes is left unconstrained. Concentrated loads, which were obtained as reaction forces (at the bottom nodes of the cylinder) for the deformation state in Step 1, are applied on the nodes on top of the cylinder.

Analytical solution: The deformation state should be identical to that obtained in Step 1.

Step 4 (perturbation):

Loading and boundary conditions: All nodes are fixed in the circumferential direction. In addition, all nodes at the bottom of the cylinder are fixed in the axial direction. The radial motion of all nodes is left

unconstrained. A distributed pressure load of magnitude 500 is applied to the inner surface, thereby expanding the cylinder uniformly.

Analytical solution: The hoop stress is 10000.

Step 5 (perturbation):

Loading and boundary conditions: All nodes are fixed in the circumferential direction. In addition, all nodes at the bottom of the cylinder are fixed in the axial direction. The radial motion of all nodes is left unconstrained. The temperature of all nodes is prescribed to be 5000, leading to thermal strains.

Analytical solution: $\epsilon_{11} = \epsilon_{22} = 0.0005$.

History definition 4 (for all element types; uses a local coordinate system)

Step 1 (geometrically nonlinear):

Loading: All degrees of freedom at all nodes are constrained. This step is recommended to apply the initial stresses. In subsequent steps all the necessary boundary conditions are applied.

Step 2 (geometrically nonlinear):

Loading: A concentrated load of magnitude 2 is applied to the top of the cylinder. To facilitate the application of the load, a beam-type multi-point constraint is used to connect the nodes on top of the cylinder to a main node.

Results and discussion

History definition 1:

All elements yield solutions that are very close to the analytical solutions.

History definition 2:

The solutions are very close to the state obtained at the end of Step 3 in history definition 1.

History definition 3:

All elements yield solutions that are very close to the analytical solutions.

History definition 4:

The results are compared with those from a similar model using an MGAX1 (axisymmetric membrane elements that support twist) element. The results match very well.

Input files

emc6srs3.inp	MCL6 elements using history definition 1.
emc9srs3.inp	MCL9 elements using history definition 1.
emc6srs4.inp	MCL6 elements using history definition 2.
emc9srs4.inp	MCL9 elements using history definition 2.

emc6srp3.inp	MCL6 elements using history definition 3.
emc9srp3.inp	MCL9 elements using history definition 3.
emc6sro3.inp	MCL6 elements with **ORIENTATION* using history definition 4.
emc9sro3.inp	MCL9 elements with **ORIENTATION* using history definition 4.

Verification of beam elements and section types

Products: Abaqus/Standard

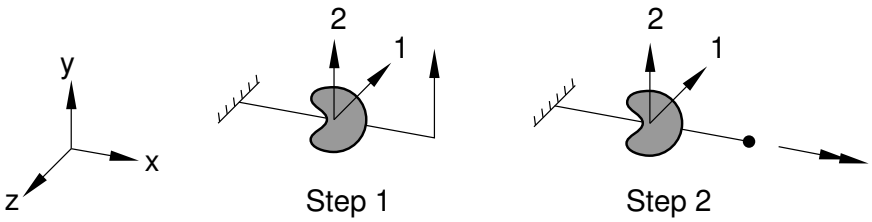
Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

B21H B22 B23H B31 B31OS B31OSH B32H B32OS B32OSH B33H
PIPE21H PIPE22 PIPE31H PIPE32
ELBOW31 ELBOW31B ELBOW31C ELBOW32

Problem description



The problem consists of a cantilever beam lying along the *x*-axis. The length of the beam is 75.0, and the model is made up of five elements. For two-dimensional elements, the problem consists of one step in which a transverse load of 25.0 is applied to the end of the beam. For three-dimensional elements this is followed by an additional step in which a moment of 25.0 is applied around the *x*-axis. Numerous tests with similar geometries and loadings are run to test the available options associated with each of the section definitions. Local coordinate directions are requested in the input files with the open thin-walled slit ring sections (*eb3ia3sd.inp*, *eb3ja3sd.inp*, *ebo3a3sd.inp*) and in two input files using a general beam section (*eb32gssd.inp*, *eb3jgssd.inp*).

Material: Linear elastic, Young's modulus = 3.0×10^6, Poisson's ratio = 0.3.

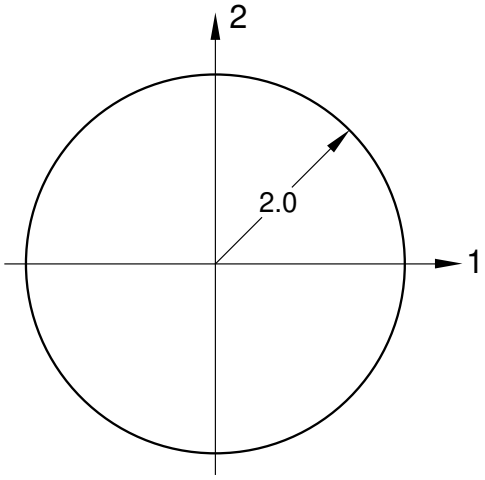
Section types: Arbitrary (Open and Closed), Box, Channel, Circular, Elbow, General, Hat, Hexagonal, I, L, Nonlinear General, Pipe, Rectangular, Trapezoidal.

Section forces

All problems are statically determinate, and section forces have been verified to be correct.

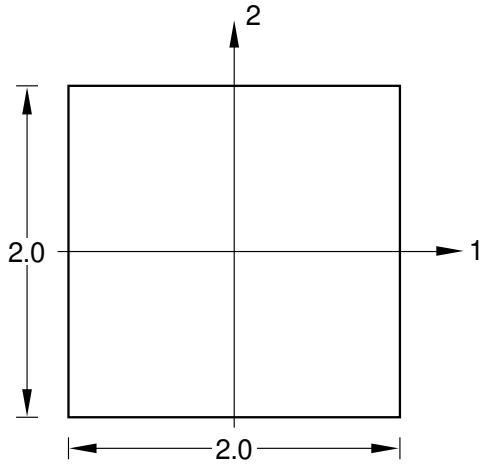
Reference solution

Solid circular sections:



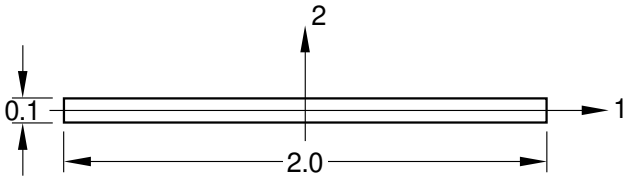
$u_2 = 9.325 \times 10^{-2}$, $\phi_3 = 1.865 \times 10^{-3}$ (Step 1), $\phi_1 = 6.466 \times 10^{-5}$ (Step 2)

Solid square sections:



$u_2 = .8789$, $\phi_3 = 1.758 \times 10^{-2}$ (Step 1), $\phi_1 = 7.224 \times 10^{-4}$ (Step 2)

Solid thin rectangular sections:

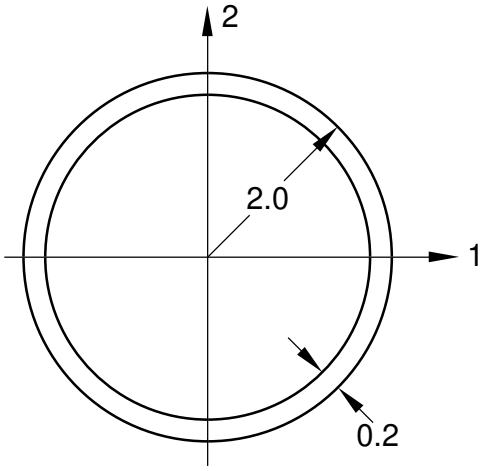


$u_2 = .7031$, $\phi_3 = 1.406 \times 10^{-2}$ (Step 1), $\phi_1 = 2.440 \times 10^{-4}$ (Step 2)

Note:

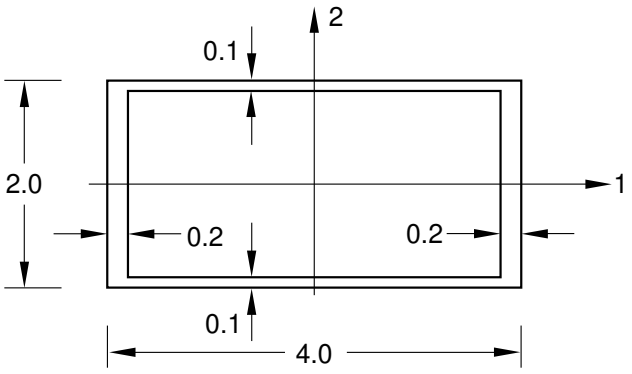
Loadings changed to .0025.

Closed thin-walled pipe sections:



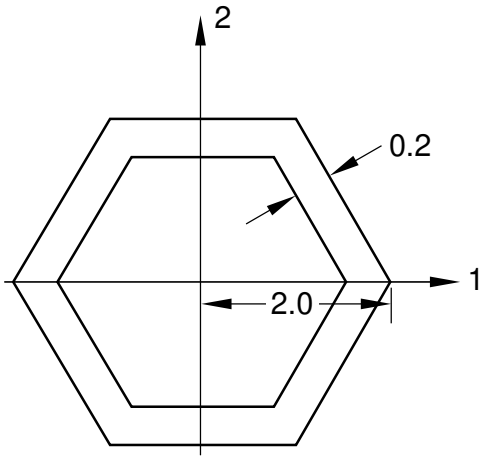
$u_2 = .2712$, $\phi_3 = 5.423 \times 10^{-3}$ (Step 1), $\phi_1 = 1.885 \times 10^{-4}$ (Step 2)

Closed thin-walled box sections:



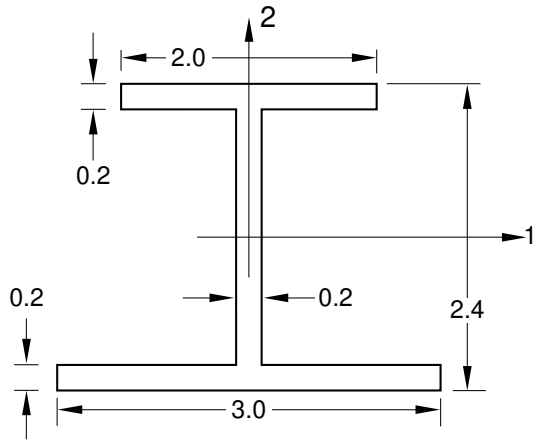
$u_2 = 1.278$, $\phi_3 = 2.556 \times 10^{-2}$ (Step 1), $\phi_1 = 7.521 \times 10^{-4}$ (Step 2)

Closed thin-walled hexagonal sections:



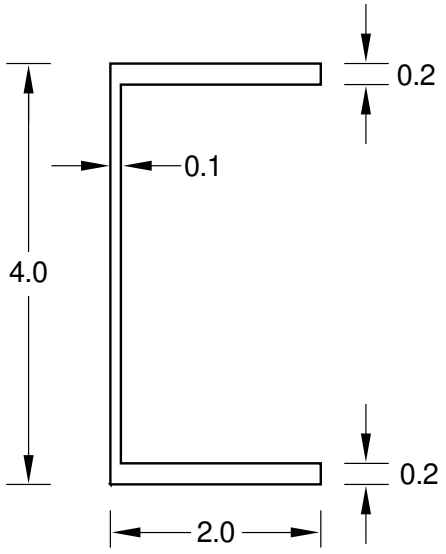
$u_2 = .3489$, $\phi_3 = 6.978 \times 10^{-3}$ (Step 1), $\phi_1 = 2.796 \times 10^{-4}$ (Step 2)

Open thin-walled I-sections:



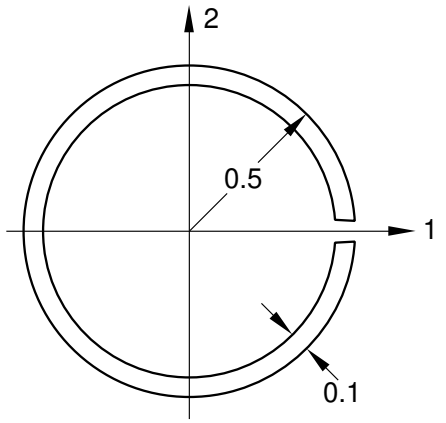
$u_2 = .8931$, $\phi_3 = 1.786 \times 10^{-2}$ (Step 1), $\phi_1 = 7.537 \times 10^{-2}$ (Step 2)

Open thin-walled C-sections:



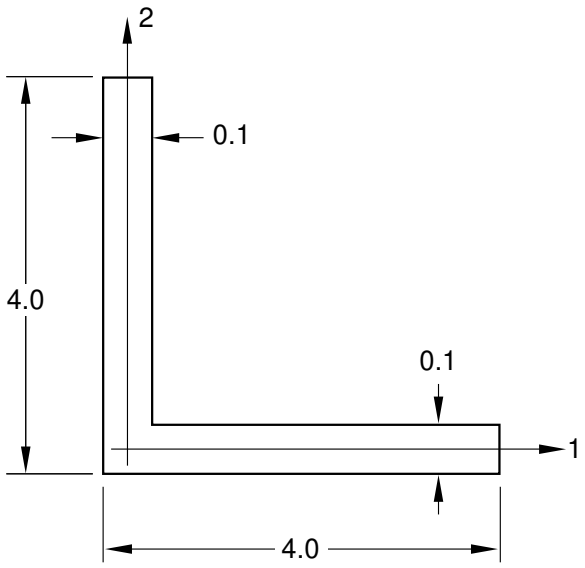
$u_2 = .3564$, $\phi_3 = 7.112 \times 10^{-3}$ (Step 1), $\phi_1 = .1081$ (Step 2)

Open thin-walled slit ring sections:



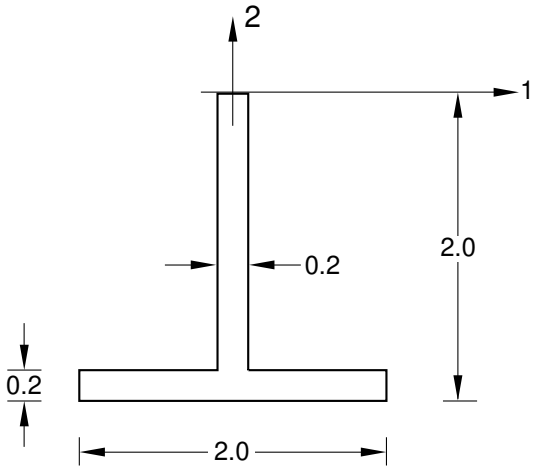
$u_2 = 29.84$, $\phi_3 = .5949$ (Step 1), $\phi_1 = 1.388$ (Step 2)

Open thin-walled L-sections:



$u_2 = 1.422$, $\phi_3 = 2.857 \times 10^{-2}$ (Step 1), $\phi_1 = .6177$ (Step 2)

Open thin-walled T-sections:



$u_2 = 4.069$, $\phi_3 = 8.138 \times 10^{-2}$ (Step 1), $\phi_1 = .1563$ (Step 2)

Results and discussion

The results for Step 1 are within 1% of the analytical values.

The results for Step 2 are less accurate for ϕ. While results for rods and cylinders are exact, other closed sections can have an error of several percent. For open section beams ϕ can have errors in excess of 10% unless an open section beam element type is used. Open section beams include warping, which can have a significant effect on ϕ.

Input files

Arbitrary cross-section, beam section tests:

eb32a1sd.inp	B31 elements, channel-origin at shear center.
eb32a2sd.inp	B31 elements, channel-origin not at shear center.
eb32a2osd.inp	B31 elements, channel-origin at shear center with **BEAM SECTION OFFSET*.
eb32a3sd.inp	B31 elements, slit circular section.
eb32absd.inp	B31 elements, box section.
eb32aisd.inp	B31 elements, I-section.
eb32alsd.inp	B31 elements, L-section.
eb3ia1sd.inp	B32H elements, channel-origin at shear center.
eb3ia2sd.inp	B32H elements, channel-origin not at shear center.
eb3ia2osd.inp	B32H elements, channel-origin at shear center with **BEAM SECTION OFFSET*.
eb3ia3sd.inp	B32H elements, slit circular section.
eb3iabsd.inp	B32H elements, box section.
eb3iaisd.inp	B32H elements, I-section.
eb3ialsd.inp	B32H elements, L-section.
eb3ja1sd.inp	B33H elements, channel-origin at shear center.
eb3ja2sd.inp	B33H elements, channel-origin not at shear center.
eb3ja2osd.inp	B33H elements, channel-origin at shear center with **BEAM SECTION OFFSET*.
eb3ja3sd.inp	B33H elements, slit circular section.
eb3jabsd.inp	B33H elements, box section.
eb3jaisd.inp	B33H elements, I-section.
eb3jalsd.inp	B33H elements, L-section.
ebo2a1sd.inp	B31OS elements, channel-origin at shear center.
ebo2a2sd.inp	B31OS elements, channel-origin not at shear center.
ebo2a3sd.inp	B31OS elements, slit circular section.
ebo2aisd.inp	B31OS elements, I-section.
ebo2alsd.inp	B31OS elements, L-section.
ebo3a1sd.inp	B32OS elements, channel-origin at shear center.
ebo3a2sd.inp	B32OS elements, channel-origin not at shear center.
ebo3a3sd.inp	B32OS elements, slit circular section.
ebo3aisd.inp	B32OS elements, I-section.
ebo3alsd.inp	B32OS elements, L-section.
eboha1sd.inp	B31OSH elements, channel-origin at shear center.
eboha2sd.inp	B31OSH elements, channel-origin not at shear center.
eboha3sd.inp	B31OSH elements, slit circular section.
ebohaisd.inp	B31OSH elements, I-section.
ebohalsd.inp	B31OSH elements, L-section.
eboia1sd.inp	B32OSH elements, channel-origin at shear center.
eboia2sd.inp	B32OSH elements, channel-origin not at shear center.

eboia3sd.inp	B32OSH elements, slit circular section.
eboiaisd.inp	B32OSH elements, I-section.
eboialsd.inp	B32OSH elements, L-section.

Arbitrary cross-section, general beam section tests:

eb32d1sd.inp	B31 elements, channel-origin at shear center.
eb32d2sd.inp	B31 elements, channel-origin not at shear center.
eb32d3sd.inp	B31 elements, slit circular section.
eb32dbsd.inp	B31 elements, box section.
eb32disd.inp	B31 elements, I-section.
eb32dlsd.inp	B31 elements, L-section.
eb3id1sd.inp	B32H elements, channel-origin at shear center.
eb3id2sd.inp	B32H elements, channel-origin not at shear center.
eb3id3sd.inp	B32H elements, slit circular section.
eb3idbsd.inp	B32H elements, box section.
eb3idisd.inp	B32H elements, I-section.
eb3idlsd.inp	B32H elements, L-section.
eb3jd1sd.inp	B33H elements, channel-origin at shear center.
eb3jd2sd.inp	B33H elements, channel-origin not at shear center.
eb3jd3sd.inp	B33H elements, slit circular section.
eb3jdbsd.inp	B33H elements, box section.
eb3jdisd.inp	B33H elements, I-section.
eb3jdlsd.inp	B33H elements, L-section.
ebo2d1sd.inp	B31OS elements, channel-origin at shear center.
ebo2d2sd.inp	B31OS elements, channel-origin not at shear center.
ebo2d3sd.inp	B31OS elements, slit circular section.
ebo2disd.inp	B31OS elements, I-section.
ebo2dlsd.inp	B31OS elements, L-section.
ebo3d1sd.inp	B32OS elements, channel-origin at shear center.
ebo3d2sd.inp	B32OS elements, channel-origin not at shear center.
ebo3d3sd.inp	B32OS elements, slit circular section.
ebo3disd.inp	B32OS elements, I-section.
ebo3dlsd.inp	B32OS elements, L-section.
ebohd1sd.inp	B31OSH elements, channel-origin at shear center.
ebohd2sd.inp	B31OSH elements, channel-origin not at shear center.
ebohd2osd.inp	B31OSH elements, channel-origin at shear center with **BEAM SECTION OFFSET*.
ebohd3sd.inp	B31OSH elements, slit circular section.
ebohdisd.inp	B31OSH elements, I-section.
ebohdlsd.inp	B31OSH elements, L-section.
eboid1sd.inp	B32OSH elements, channel-origin at shear center.
eboid2sd.inp	B32OSH elements, channel-origin not at shear center.
eboid2osd.inp	B32OSH elements, channel-origin at shear center with **BEAM SECTION OFFSET*.

eboid3sd.inp	B32OSH elements, slit circular section.
eboidisd.inp	B32OSH elements, I-section.
eboidlsd.inp	B32OSH elements, L-section.

Box cross-section, beam section tests:

eb23bdsd.inp	B22 elements, default integration.
eb23bnsd.inp	B22 elements, nondefault integration.
eb2hbdsd.inp	B21H elements, default integration.
eb2hbnsd.inp	B21H elements, nondefault integration.
eb2jbdsd.inp	B23H elements, default integration.
eb2jbnsd.inp	B23H elements, nondefault integration.
eb32bdsd.inp	B31 elements, default integration.
eb32bnsd.inp	B31 elements, nondefault integration.
eb3ibdsd.inp	B32H elements, default integration.
eb3ibnsd.inp	B32H elements, nondefault integration.
eb3jbdsd.inp	B33H elements, default integration.
eb3jbnsd.inp	B33H elements, nondefault integration.

Box cross-section, general beam section tests:

eb23exsd.inp	B22 elements.
eb2hexsd.inp	B21H elements.
eb2jexsd.inp	B23H elements.
eb32exsd.inp	B31 elements.
eb3iexsd.inp	B32H elements.
eb3jexsd.inp	B33H elements.

Channel cross-section, general beam section tests:

channel_b31.inp	B31 elements, channel-origin not at shear center.
channel_b32h.inp	B32H elements, channel-origin not at shear center.
channel_b33h.inp	B33H elements, channel-origin not at shear center.
channel_b31os.inp	B31OS elements, channel-origin not at shear center.
channel_b31os_offset.inp	B31OS elements, channel-origin at shear center with **BEAM SECTION OFFSET*.
channel_b32os.inp	B32OS elements, channel-origin not at shear center.
channel_b32os_offset.inp	B32OS elements, channel-origin at shear center with **BEAM SECTION OFFSET*.
channel_b31osh.inp	B31OSH elements, channel-origin not at shear center.
channel_b32osh.inp	B32OSH elements, channel-origin not at shear center.

Circular cross-section, beam section tests:

eb23cdsd.inp	B22 elements, default integration.
eb23cnsd.inp	B22 elements, nondefault integration.
eb2hcdsd.inp	B21H elements, default integration.

eb2hcnsd.inp	B21H elements, nondefault integration.
eb2jcdsd.inp	B23H elements, default integration.
eb2jcnsd.inp	B23H elements, nondefault integration.
eb32cdsd.inp	B31 elements, default integration.
eb32cnsd.inp	B31 elements, nondefault integration.
eb3icdsd.inp	B32H elements, default integration.
eb3icnsd.inp	B32H elements, nondefault integration.
eb3jcdsd.inp	B33H elements, default integration.
eb3jcnsd.inp	B33H elements, nondefault integration.

Circular cross-section, general beam section tests:

eb23fxsd.inp	B22 elements.
eb2hfxsd.inp	B21H elements.
eb2jfxsd.inp	B23H elements.
eb32fxsd.inp	B31 elements.
eb3ifxsd.inp	B32H elements.
eb3jfxsd.inp	B33H elements.

General cross-section tests:

eb23gpsd.inp	B22 elements, pipe section.
eb23gssd.inp	B22 elements, solid square section.
eb2hgpsd.inp	B21H elements, pipe section.
eb2hgssd.inp	B21H elements, solid square section.
eb2jgpsd.inp	B23H elements, pipe section.
eb2jgssd.inp	B23H elements, solid square section.
eb32gisd.inp	B31 elements, I-section.
eb32gpsd.inp	B31 elements, pipe section.
eb32gssd.inp	B31 elements, solid square section.
eb3igpsd.inp	B32H elements, pipe section.
eb3igssd.inp	B32H elements, solid square section.
eb3jgpsd.inp	B33H elements, pipe section.
eb3jgssd.inp	B33H elements, solid square section.
ebo2gisd.inp	B31OS elements, I-section.
ebo3gisd.inp	B32OS elements, I-section.
ebohgisd.inp	B31OSH elements, I-section.
ebohgisd.inp	B32OSH elements, I-section.

Hat cross-section, general beam section tests:

hat_b31.inp	B31 elements, hat-origin not at shear center.
hat_b32h.inp	B32H elements, hat-origin not at shear center.
hat_b33h.inp	B33H elements, hat-origin not at shear center.
hat_b31os.inp	B31OS elements, hat-origin not at shear center.
hat_b32os.inp	B32OS elements, hat-origin not at shear center.

hat_b31osh.inp	B31OSH elements, hat-origin not at shear center.
hat_b32osh.inp	B32OSH elements, hat-origin not at shear center.

Hexagonal cross-section, beam section tests:

eb23hdsd.inp	B22 elements, default integration.
eb23hnsd.inp	B22 elements, nondefault integration.
eb2hhdsd.inp	B21H elements, default integration.
eb2hhnsd.inp	B21H elements, nondefault integration.
eb2jhdsd.inp	B23H elements, default integration.
eb2jhnsd.inp	B23H elements, nondefault integration.
eb32hdsd.inp	B31 elements, default integration.
eb32hnsd.inp	B31 elements, nondefault integration.
eb3ihdsd.inp	B32H elements, default integration.
eb3ihnsd.inp	B32H elements, nondefault integration.
eb3jhdsd.inp	B33H elements, default integration.
eb3jhnsd.inp	B33H elements, nondefault integration.

Hexagonal cross-section, general beam section tests:

eb23jxsd.inp	B22 elements.
eb2hjxsd.inp	B21H elements.
eb2jjxsd.inp	B23H elements.
eb32jxsd.inp	B31 elements.
eb3ijxsd.inp	B32H elements.
eb3jjxsd.inp	B33H elements.

I cross-section, beam section tests:

eb23idsd.inp	B22 elements, default integration.
eb23insd.inp	B22 elements, nondefault integration.
eb2hidsd.inp	B21H elements, default integration.
eb2hinsd.inp	B21H elements, nondefault integration.
eb2jidsd.inp	B23H elements, default integration.
eb2jinsd.inp	B23H elements, nondefault integration.
eb32idsd.inp	B31 elements, default integration.
eb32insd.inp	B31 elements, nondefault integration.
eb32itsd.inp	B31 elements, T-section.
eb3iidsd.inp	B32H elements, default integration.
eb3iinsd.inp	B32H elements, nondefault integration.
eb3iitsd.inp	B32H elements, T-section.
eb3jidsd.inp	B33H elements, default integration.
eb3jinsd.inp	B33H elements, nondefault integration.
eb3jitsd.inp	B33H elements, T-section.
ebo2idsd.inp	B31OS elements, default integration.
ebo2insd.inp	B31OS elements, nondefault integration.
ebo2itsd.inp	B31OS elements, T-section.

ebo3idsd.inp	B32OS elements, default integration.
ebo3insd.inp	B32OS elements, nondefault integration.
ebo3itsd.inp	B32OS elements, T-section.
ebohidsd.inp	B31OSH elements, default integration.
ebohinsd.inp	B31OSH elements, nondefault integration.
ebohitsd.inp	B31OSH elements, T-section.
eboiidsd.inp	B32OSH elements, default integration.
eboiinsd.inp	B32OSH elements, nondefault integration.
eboiitsd.inp	B32OSH elements, T-section.

I cross-section, general beam section tests:

eb23kxsd.inp	B22 elements.
eb2hkxsd.inp	B21H elements.
eb2jkxsd.inp	B23H elements.
eb32ktsd.inp	B31 elements, T-section.
eb32kxsd.inp	B31 elements.
eb3iktsd.inp	B32H elements, T-section.
eb3ikxsd.inp	B32H elements.
eb3jktsd.inp	B33H elements. T-section.
eb3jkxsd.inp	B33H elements.
ebo2ktsd.inp	B31OS elements, T-section.
ebo2kxsd.inp	B31OS elements.
ebo3ktsd.inp	B32OS elements, T-section.
ebo3kxsd.inp	B32OS elements.
ebohktsd.inp	B31OSH elements, T-section.
ebohkxsd.inp	B31OSH elements.
eboiktsd.inp	B32OSH elements, T-section.
eboikxsd.inp	B32OSH elements.

L cross-section, beam section tests:

eb32ldsd.inp	B31 elements, default integration.
eb32lnsd.inp	B31 elements, nondefault integration.
eb3ildsd.inp	B32H elements, default integration.
eb3ilnsd.inp	B32H elements, nondefault integration.
eb3jldsd.inp	B33H elements, default integration.
eb3jlnsd.inp	B33H elements, nondefault integration.
ebo2ldsd.inp	B31OS elements, default integration.
ebo2lnsd.inp	B31OS elements, nondefault integration.
ebo3ldsd.inp	B32OS elements, default integration.
ebo3lnsd.inp	B32OS elements, nondefault integration.
ebohldsd.inp	B31OSH elements, default integration.
ebohlnsd.inp	B31OSH elements, nondefault integration.
eboildsd.inp	B32OSH elements, default integration.
eboilnsd.inp	B32OSH elements, nondefault integration.

L cross-section, general beam section tests:

eb32mxsd.inp	B31 elements.
eb3imxsd.inp	B32H elements.
eb3jmxsd.inp	B33H elements.
ebo2mxsd.inp	B31OS elements.
ebo3mxsd.inp	B32OS elements.
ebo3mxsd_mat.inp	B32OS elements, **BEAM GENERAL SECTION* with MATERIAL parameter.
ebohmxsd.inp	B31OSH elements.
eboimxsd.inp	B32OSH elements.

Nonlinear general cross-section tests:

eb23ncsd.inp	B22 elements, circular section.
eb23nvsd.inp	B22 elements, section data defined using a table of values.
eb2hncsd.inp	B21H elements, circular section.
eb2hnvsd.inp	B21H elements, section data defined using a table of values.
eb2jncsd.inp	B23H elements, circular section.
eb2jnvsd.inp	B23H elements, section data defined using a table of values.
eb32ncsd.inp	B31 elements, circular section.
eb32nvsd.inp	B31 elements, section data defined using a table of values.
eb3incsd.inp	B32H elements, circular section.
eb3invsd.inp	B32H elements, section data defined using a table of values.
eb3jncsd.inp	B33H elements, circular section.
eb3jnvsd.inp	B33H elements, section data defined using a table of values.

Pipe cross-section, beam section tests:

eb23pdsd.inp	B22 elements, default integration.
eb23pnsd.inp	B22 elements, nondefault integration.
eb2hpdsd.inp	B21H elements, default integration.
eb2hpnsd.inp	B21H elements, nondefault integration.
eb2jpdsd.inp	B23H elements, default integration.
eb2jpnsd.inp	B23H elements, nondefault integration.
eb32pdsd.inp	B31 elements, default integration.
eb32pnsd.inp	B31 elements, nondefault integration.
eb3ipdsd.inp	B32H elements, default integration.
eb3ipnsd.inp	B32H elements, nondefault integration.
eb3jpdsd.inp	B33H elements, default integration.
eb3jpnsd.inp	B33H elements, nondefault integration.
ep23pdsd.inp	PIPE22 elements, default integration.
ep23pnsd.inp	PIPE22 elements, nondefault integration.
ep2hpdsd.inp	PIPE21H elements, default integration.
ep2hpnsd.inp	PIPE21H elements, nondefault integration.
ep33pdsd.inp	PIPE32 elements, default integration.
ep33pnsd.inp	PIPE32 elements, nondefault integration.

ep3hpdsd.inp	PIPE31H elements, default integration.
ep3hpnsd.inp	PIPE31H elements, nondefault integration.

Pipe cross-section, general beam section tests:

eb23oxsd.inp	B22 elements.
eb2hoxsd.inp	B21H elements.
eb2joxsd.inp	B23H elements.
eb32oxsd.inp	B31 elements.
eb3ioxsd.inp	B32H elements.
eb3joxsd.inp	B33H elements.

Rectangular cross-section, beam section tests:

eb23rssd.inp	B22 elements, solid square section, default integration.
eb23r4sd.inp	B22 elements, solid square section, nondefault integration.
eb23rrsd.inp	B22 elements, thin rectangular section.
eb23r5sd.inp	B22 elements, thin rectangular section, nondefault integration.
eb2hrssd.inp	B21H elements, solid square section, default integration.
eb2hr4sd.inp	B21H elements, solid square section, nondefault integration.
eb2hrrsd.inp	B21H elements, thin rectangular section.
eb2hr5sd.inp	B21H elements, thin rectangular section, nondefault integration.
eb2jrssd.inp	B23H elements, solid square section, default integration.
eb2jr4sd.inp	B23H elements, solid square section, nondefault integration.
eb2jrrsd.inp	B23H elements, thin rectangular section.
eb2jr5sd.inp	B23H elements, thin rectangular section, nondefault integration.
eb32rssd.inp	B31 elements, solid square section, default integration.
eb32r4sd.inp	B31 elements, solid square section, nondefault integration.
eb32rrsd.inp	B31 elements, thin rectangular section.
eb32r5sd.inp	B31 elements, thin rectangular section, nondefault integration.
eb3irssd.inp	B32H elements, solid square section, default integration.
eb3ir4sd.inp	B32H elements, solid square section, nondefault integration.
eb3irrsd.inp	B32H elements, thin rectangular section.
eb3ir5sd.inp	B32H elements, thin rectangular section, nondefault integration.
eb3jrssd.inp	B33H elements, solid square section, default integration.
eb3jr4sd.inp	B33H elements, solid square section, nondefault integration.
eb3jrrsd.inp	B33H elements, thin rectangular section.
eb3jr5sd.inp	B33H elements, thin rectangular section, nondefault integration.

Rectangular cross-section, general beam section tests:

eb23qrsd.inp	B22 elements, thin rectangular section.
eb23qssd.inp	B22 elements, solid square section, default integration.
eb2hqrsd.inp	B21H elements, thin rectangular section.
eb2hqssd.inp	B21H elements, solid square section, default integration.
eb2jqrsd.inp	B23H elements, thin rectangular section.
eb2jqssd.inp	B23H elements, solid square section, default integration.
eb32qrsd.inp	B31 elements, thin rectangular section.

eb32qssd.inp	B31 elements, solid square section, default integration.
eb3iqrsd.inp	B32H elements, thin rectangular section.
eb3iqssd.inp	B32H elements, solid square section, default integration.
eb3jqrsd.inp	B33H elements, thin rectangular section.
eb3jqssd.inp	B33H elements, solid square section, default integration.

Trapezoidal cross-section, beam section tests:

eb23t4sd.inp	B22 elements, solid square section, nondefault integration.
eb23t5sd.inp	B22 elements, thin rectangular section, nondefault integration.
eb23t6sd.inp	B22 elements, solid square section, nondefault local origin.
eb23t6osd.inp	B22 elements, solid square section, nondefault local origin with **BEAM SECTION OFFSET*.
eb23trsd.inp	B22 elements, thin rectangular section.
eb23tssd.inp	B22 elements, solid square section, default integration.
eb2ht4sd.inp	B21H elements, solid square section, nondefault integration.
eb2ht5sd.inp	B21H elements, thin rectangular section, nondefault integration.
eb2ht6sd.inp	B21H elements, solid square section, nondefault local origin.
eb2ht6osd.inp	B21H elements, solid square section, nondefault local origin with **BEAM SECTION OFFSET*.
eb2htrsd.inp	B21H elements, thin rectangular section.
eb2htssd.inp	B21H elements, solid square section, default integration.
eb2jt4sd.inp	B23H elements, solid square section, nondefault integration.
eb2jt5sd.inp	B23H elements, thin rectangular section, nondefault integration.
eb2jt6sd.inp	B23H elements, solid square section, nondefault local origin.
eb2jt6osd.inp	B23H elements, solid square section, nondefault local origin with **BEAM SECTION OFFSET*.
eb2jtrsd.inp	B23H elements, thin rectangular section.
eb2jtssd.inp	B23H elements, solid square section, default integration.
eb32t4sd.inp	B31 elements, solid square section, nondefault integration.
eb32t5sd.inp	B31 elements, thin rectangular section, nondefault integration.
eb32t6sd.inp	B31 elements, solid square section, nondefault local origin.
eb32trsd.inp	B31 elements, thin rectangular section.
eb32tssd.inp	B31 elements, solid square section, default integration.
eb3it4sd.inp	B32H elements, solid square section, nondefault integration.
eb3it5sd.inp	B32H elements, thin rectangular section, nondefault integration.
eb3it6sd.inp	B32H elements, solid square section, nondefault local origin.
eb3itrsd.inp	B32H elements, thin rectangular section.
eb3itssd.inp	B32H elements, solid square section, default integration.
eb3jt4sd.inp	B33H elements, solid square section, nondefault integration.
eb3jt5sd.inp	B33H elements, thin rectangular section, nondefault integration.
eb3jt6sd.inp	B33H elements, solid square section, nondefault local origin.
eb3jtrsd.inp	B33H elements, thin rectangular section.
eb3jtssd.inp	B33H elements, solid square section, default integration.

Trapezoidal cross-section, general beam section tests:

eb23s6sd.inp	B22 elements, solid square section, nondefault local origin.
eb23srsd.inp	B22 elements, thin rectangular section.
eb23sssd.inp	B22 elements, solid square section, default integration.
eb2hs6sd.inp	B21H elements, solid square section, nondefault local origin.
eb2hsrsd.inp	B21H elements, thin rectangular section.
eb2hsssd.inp	B21H elements, solid square section, default integration.
eb2js6sd.inp	B23H elements, solid square section, nondefault local origin.
eb2jsrsd.inp	B23H elements, thin rectangular section.
eb2jsssd.inp	B23H elements, solid square section, default integration.
eb32s6sd.inp	B31 elements, solid square section, nondefault local origin.
eb32srsd.inp	B31 elements, thin rectangular section.
eb32sssd.inp	B31 elements, solid square section, default integration.
eb3is6sd.inp	B32H elements, solid square section, nondefault local origin.
eb3isrsd.inp	B32H elements, thin rectangular section.
eb3isssd.inp	B32H elements, solid square section, default integration.
eb3js6sd.inp	B33H elements, solid square section, nondefault local origin.
eb3jsrsd.inp	B33H elements, thin rectangular section.
eb3jsssd.inp	B33H elements, solid square section, default integration.

Reference solutions:

erefscsd.inp	Reference solution for the circular section beams.
erefsisd.inp	Reference solution for the I-section beams.
erefslsd.inp	Reference solution for the L-section beams.
erefstsd.inp	Reference solution for the T-section beams.

Beam added inertia

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Features tested

This section provides basic verification tests for beams with additional inertia, which can be used with all Timoshenko beams. In Abaqus/Standard it also verifies the isotropic versus the exact rotary inertia formulation for Timoshenko beams.

Verification tests for Timoshenko beams in Abaqus/Standard

Elements tested

PIPE21 PIPE21H PIPE22 PIPE22H PIPE31 PIPE31H PIPE32 PIPE32H
B21 B21H B22 B22H B31 B31H B31OS B31OSH B32 B32H B32OS B32OSH

Problem description

There are two sets of problems presented in this section. The first set includes four input files: *b31_dyn_iso.inp*, *b31_dyn_exact.inp*, *b31_moddyn_iso.inp*, and *b31_moddyn_exact.inp*. These analyses compare the dynamic response to an acceleration record on a single-element cantilever structure made of B31 elements using the isotropic or exact rotary inertia formulation. Comparisons are made between the direct-integration implicit dynamic and the modal dynamic procedures. To change the rotary inertia formulation for Timoshenko beams, isotropic rotary inertia or exact (default) rotary inertia is used for the beam or beam general sections.

The second set of problems verifies the beam with the additional inertia procedure. This procedure allows adding mass and rotary inertia properties per element length at specified locations on the beam cross-section. The beam's mass together with the added mass may combine to give an offset between the location of the node and the center of mass for the cross-section. That offset produces the coupling between the translational degrees of freedom and the rotational degrees of freedom in the mass matrix for the element. A pair of input files, *xbeamaddinertia_std_lin3d.inp* and *xbeamaddinertia_std_quad3d.inp*, shows the concept of the offset mass for the beam element that can also be modeled with MASS and ROTARYI elements with appropriate BEAM-type MPC definitions to accommodate the mass offset. The remaining single-element input files verify various cross-section types for transient dynamic and eigenvalue extraction procedures. Input files *pmcp_pipe2d_bai.inp*, *pmcp_beam2d_bai.inp*, *pmcp_pipe3d_bai.inp*, and *pmcp_beam3d_bai.inp* are collections of all pipe and all beam elements placed in a plane or space. The beam with the additional inertia procedure is used for all beam section definitions. These multiple step analyses verify the frequency, static (with mass-dependent loads), steady-state (mode-based and direct), modal dynamic, and direct-integration implicit dynamic procedures.

Results and discussion

The results compare well with the concentrated masses and rotary inertia element models and differ from the isotropic formulation as predicted.

Input files

b31_dyn_iso.inp	B31 element, transient dynamic, isotropic rotary inertia formualtion.

b31_moddyn_iso.inp	B31 element, modal dynamic, isotropic rotary inertia formulation.
b31_dyn_exact.inp	B31 element, transient dynamic, exact rotary inertia formulation.
b31_moddyn_exact.inp	B31 element, modal dynamic, exact rotary inertia formulation.
b21_circ_bai_45.inp	B21 element with circular cross-section, transient dynamic.
b22_rect_bai_freq.inp	B22 element with rectangular cross-section, frequency extraction.
b21h_circ_bai_freq.inp	B21H element with circular cross-section, frequency extraction.
b31_circ_bai.inp	B31 element with circular cross-section, transient dynamic, and unsymmetric solver.
b31_circ_mass_ri.inp	MASS and ROTARYI elements, transient dynamic.
b32_box_bai.inp	B32 with box cross-section, transient dynamic.
b31os_i_bai.inp	B31OS element with I cross-section, transient dynamic procedure.
pipe31h_bai_45.inp	PIPE31H, transient dynamic.
pmcp_beam2d_bai.inp	All two-dimensional beams, various dynamic procedures.
pmcp_beam3d_bai.inp	All three-dimensional beams, various dynamic procedures.
pmcp_pipe2d_bai.inp	All two-dimensional pipe elements, various dynamic procedures.
pmcp_pipe3d_bai.inp	All three-dimensional pipe elements, various dynamic procedures.

Explicit dynamic test of beams with additional inertia

Elements tested

B21 B22 B31 B32

Problem description

This problem verifies the use of beams with additional inertia in Abaqus/Explicit. Identical beam elements are assigned additional mass and rotary inertia in two ways: using the beams with the additional inertia procedure and by defining additional point mass and rotary inertia elements and rigidly constraining them to the beam nodes using BEAM-type MPCs. The solutions obtained using the two methods are compared. Four cases, each comprising one of the four beam element types available in Abaqus/Explicit, are considered.

For each case four beam elements with the same element length are defined. Two of the beam elements are assigned identical section properties using the beam section procedure, and the remaining two are assigned identical section properties using the general beam section procedure.

One of the elements with section properties given by the beam section procedure has additional mass and rotary inertia assigned to it using the beam with the additional inertia procedure. For the second beam element with a beam section, additional nodes are defined at locations offset from the element nodes and MASS and ROTARYI elements are defined at the offset nodes. BEAM-type MPCs connect each node of the second beam to its corresponding offset node. The offset node corresponding to each node of the second beam lies in the cross-section passing through the beam node and has the same local coordinates with respect to the beam node as the center of mass coordinates defined for the first beam. Similarly, the mass and inertia assigned to the offset nodes are exactly equivalent to those assigned to the first beam element using the beam with the additional inertia procedure.

The two beam elements with general beam sections are also subjected to the same test. One of them is assigned additional mass and inertia, while the other has BEAM-type MPCs connecting each node to nodal locations offset from it where MASS and ROTARYI elements with appropriate section properties are defined.

All four beams are cantilevered at one end and are subjected to the same concentrated load at the other end.

Results and discussion

On comparing the nodal displacements and rotations of each beam element with additional inertia to those of its corresponding element with BEAM-type MPCs, it is found that the nodal values match closely. This result verifies that mass and inertia values are assigned accurately for the beams with additional inertia.

Input files

xbeamaddinertia_xpl_lin2d.inp	B21 elements.
xbeamaddinertia_xpl_quad2d.inp	B22 elements.
xbeamaddinertia_xpl_lin3d.inp	B31 elements.
xbeamaddinertia_xpl_quad3d.inp	B32 elements.

Beam fluid inertia

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Features tested

This section provides basic verification tests for using fluid inertia with all Timoshenko beams.

Elements tested

B21 B22 B31 B32

Problem description

There are two sets of problems presented in this section. The first set includes the input files *b21_circ_bfi.inp* and *b21_circ_bai.inp* and verifies the use of beams with fluid inertia in a direct-integration implicit dynamic procedure. The second set consists of the remaining input files and tests the use of beams with fluid inertia in transient dynamic analysis with beam sections or general beam sections. The transient dynamic analyses are also performed using Abaqus/Explicit to verify beams with fluid inertia in Abaqus/Explicit.

Results and discussion

The results obtained from *b21_circ_bfi.inp* using beams with fluid inertia agree well with the results from *b21_circ_bai.inp*, which uses beams with additional inertia to add equivalent masses. The comparison is meaningful only for the direction in which the external excitation is applied.

The Abaqus/Explicit transient analysis results closely match the transient analysis results obtained using Abaqus/Standard.

Input files

Abaqus/Standard input files

b21_circ_bfi.inp	B21 element, frequency extraction, **BEAM FLUID INERTIA*.
b21_circ_bai.inp	B21 element, frequency extraction, **BEAM ADDED INERTIA*.
iw_bs_b21_dyl_bfi.inp	B21 element, **BEAM SECTION*, transient dynamic.
iw_bs_b22_dyl_bfi.inp	B22 element, **BEAM SECTION*, transient dynamic.
iw_bs_b31_e_dyl_bfi.inp	B31 element, **BEAM SECTION*, transient dynamic, exact rotary inertia formulation.
iw_bs_b31_i_dyl_bfi.inp	B31 element, **BEAM SECTION*, transient dynamic, isotropic rotary inertia formulation.
iw_bs_b32_e_dyl_bfi.inp	B32 element, **BEAM SECTION*, transient dynamic, exact rotary inertia formulation.
iw_bs_b32_i_dyl_bfi.inp	B32 element, **BEAM SECTION*, transient dynamic, isotropic rotary inertia formulation.

iw_bgs_b21_dyl_bfi.inp	B21 element, **BEAM GENERAL SECTION*, transient dynamic.
iw_bgs_b22_dyl_bfi.inp	B22 element, **BEAM GENERAL SECTION*, transient dynamic.
iw_bgs_b31_e_dyl_bfi.inp	B31 element, **BEAM GENERAL SECTION*, transient dynamic, exact rotary inertia formulation.
iw_bgs_b31_i_dyl_bfi.inp	B31 element, **BEAM GENERAL SECTION*, transient dynamic, isotropic rotary inertia formulation.
iw_bgs_b32_e_dyl_bfi.inp	B32 element, **BEAM GENERAL SECTION*, transient dynamic, exact rotary inertia formulation.
iw_bgs_b32_i_dyl_bfi.inp	B32 element, **BEAM GENERAL SECTION*, transient dynamic, isotropic rotary inertia formulation.
iw_bgs_b31_dyl_bfi_half.inp	B31 element, **BEAM GENERAL SECTION*, transient dynamic, the HALF parameter on the **BEAM FLUID INERTIA* option.
iw_bfi_std.inp	B31 elements, **BEAM GENERAL SECTION*, transient dynamic, different radii on the **BEAM FLUID INERTIA* option to test dynamic response to constant plane-wave pressure gradients.
iw_bfi_std_bbl.inp	B31 elements, **BEAM GENERAL SECTION*, transient dynamic, different radii on the **BEAM FLUID INERTIA* option to test dynamic response to bubble loading.
iw_bfi_std_refbbl.inp	B31 elements, **BEAM GENERAL SECTION*, transient dynamic, different radii on the **BEAM FLUID INERTIA* option to test dynamic response to bubble loading with reflections from a “hard” plane.
iw_bfi_std_refl.inp	B31 elements, **BEAM GENERAL SECTION*, transient dynamic, different radii on the **BEAM FLUID INERTIA* option to test dynamic response to spherical loading with reflections from a “hard” plane.
iw_bfi_std_reflec.inp	B31 elements, **BEAM GENERAL SECTION*, transient dynamic, different radii on the **BEAM FLUID INERTIA* option to test dynamic response to planar loading with reflections from a “soft” plane.
iw_bgsm_bfi_std_refl.inp	B31 elements, **BEAM GENERAL SECTION*, transient dynamic, comparison of meshed and general section definitions to test dynamic response to planar loading with reflections from a “hard” plane.

Abaqus/Explicit input files

iw_bs_b21_dyl_bfi_xpl.inp	B21 element, **BEAM SECTION*, transient explicit dynamic analysis.
iw_bs_b22_dyl_bfi_xpl.inp	B22 element, **BEAM SECTION*, transient explicit dynamic analysis.
iw_bs_b31_e_dyl_bfi_xpl.inp	B31 element, **BEAM SECTION*, transient explicit dynamic analysis, exact rotary inertia formulation.
iw_bs_b31_i_dyl_bfi_xpl.inp	B31 element, **BEAM SECTION*, transient explicit dynamic analysis, isotropic rotary inertia formulation.
iw_bs_b32_e_dyl_bfi_xpl.inp	B32 element, **BEAM SECTION*, transient explicit dynamic analysis, exact rotary inertia formulation.
iw_bs_b32_i_dyl_bfi_xpl.inp	B32 element, **BEAM SECTION*, transient explicit dynamic analysis, isotropic rotary inertia formulation.
iw_bgs_b21_dyl_bfi_xpl.inp	B21 element, **BEAM GENERAL SECTION*, transient explicit dynamic analysis.
iw_bgs_b22_dyl_bfi_xpl.inp	B22 element, **BEAM GENERAL SECTION*, transient explicit dynamic analysis.
iw_bgs_b31_e_dyl_bfi_xpl.inp	B31 element, **BEAM GENERAL SECTION*, transient explicit dynamic analysis, exact rotary inertia formulation.

iw_bgs_b31_i_dyl_bfi_xpl.inp	B31 element, **BEAM GENERAL SECTION*, transient explicit dynamic analysis, isotropic rotary inertia formulation.
iw_bgs_b32_e_dyl_bfi_xpl.inp	B32 element, **BEAM GENERAL SECTION*, transient explicit dynamic analysis, exact rotary inertia formulation.
iw_bgs_b32_i_dyl_bfi_xpl.inp	B32 element, **BEAM GENERAL SECTION*, transient explicit dynamic analysis, isotropic rotary inertia formulation.
iw_bgs_b31_dyl_bfi_half_xpl.inp	B31 element, **BEAM GENERAL SECTION*, transient explicit dynamic analysis, the HALF parameter on the **BEAM FLUID INERTIA* option.
iw_bfi_xpl.inp	B31 elements, **BEAM GENERAL SECTION*, transient dynamic, different radii on the **BEAM FLUID INERTIA* option to test dynamic response to constant plane-wave pressure gradients.
iw_bfi_xpl_bbl.inp	B31 elements, **BEAM GENERAL SECTION*, transient dynamic, different radii on the **BEAM FLUID INERTIA* option to test dynamic response to bubble loading.
iw_bfi_xpl_refbbl.inp	B31 elements, **BEAM GENERAL SECTION*, transient dynamic, different radii on the **BEAM FLUID INERTIA* option to test dynamic response to bubble loading with reflections from a “hard” plane.
iw_bfi_xpl_refl.inp	B31 elements, **BEAM GENERAL SECTION*, transient dynamic, different radii on the **BEAM FLUID INERTIA* option to test dynamic response to spherical loading with reflections from a “hard” plane.
iw_bfi_xpl_reflec.inp	B31 elements, **BEAM GENERAL SECTION*, transient dynamic, different radii on the **BEAM FLUID INERTIA* option to test dynamic response to planar loading with reflections from a “soft” plane.
iw_bgsm_bfi_xpl_refl.inp	B31 elements, **BEAM GENERAL SECTION*, transient dynamic, comparison of meshed and general section definitions to test dynamic response to planar loading with reflections from a “hard” plane.

Beam with end moment

Products: Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

CPE4R CPE6M C3D10M

Features tested

Concentrated loads, follower forces, multi-point constraints.

Problem description

This problem demonstrates the use of concentrated loads with CPE4R, CPE6M, and C3D10M elements in a large-strain analysis. Two beams are analyzed together. Both beams are cantilevered on one end and are subjected to a force couple (a balanced set of loads on the translation degrees of freedom) on the other end. The couple on Beam 1 (the upper beam) is comprised of follower forces, so the applied moment is independent of tip rotation. Non-follower forces generate the moment on Beam 2 (the lower beam), so the moment is a function of the tip rotation.

This problem also demonstrates a technique for introducing follower forces into a mesh generated using solid elements. A follower force in Abaqus requires a rotational degree of freedom to introduce change of direction of the application of the force. However, nodes attached to solid elements have only translational degrees of freedom. The BEAMMPC is used to activate rotational degrees of freedom at nodes where the forces are applied. The LINEARMPC is used to constrain the end of the beam to remain a plane section.

Each beam is 400 mm long (L) and 20 mm thick (h). In the finite element model all the nodes at the right side are pinned, and the nodes at the left are constrained with BEAM and LINEARMPCs so that they remain in a straight line of constant length.

The material for this problem is elastic with a constant Young's modulus of 1000 MPa and a Poisson's ratio of 0. The density is 10000 kg/m^3.

For small-strain elasticity the moment per unit width required to form a beam into a circle is given by

$$M = \frac{\pi h^3 E}{6L} = 10.47 \times 10^3 \text{ Nm/m}.$$

The force required for this moment (using the beam thickness as the moment arm) is 523.6×10^3 N. Because of dynamic effects the required forces are only 490.0×10^3 N for the CPE4R mesh, 510.0×10^3 N for the CPE6M mesh, and 4900.0 N for the C3D10M mesh. These forces are ramped on linearly over the analysis time of 0.2 seconds. The time period is chosen so that the quasi-static response can be observed with a minimum of dynamic vibration.

Results and discussion

Figure 1 shows the undeformed and deformed meshes (CPE4R) of both beams. Beam 1 forms a circle, while Beam 2 stops short of a 90° tip rotation. Since the load on Beam 2 is not a follower load, the moment arm of the force-couple decreases as the beam deflection increases. *Figure 2* shows the corresponding meshes composed

of CPE6M elements. The undeformed and deformed meshes for C3D10M elements are shown in *Figure 3*. *Figure 4* shows the time history of the tip rotations (in radians) of the two beams.

Input files

beamfollow.inp	CPE4R elements.
beamfollow_cpe6m.inp	CPE6M elements.
beamfollow_c3d10m.inp	C3D10M elements.

Figures

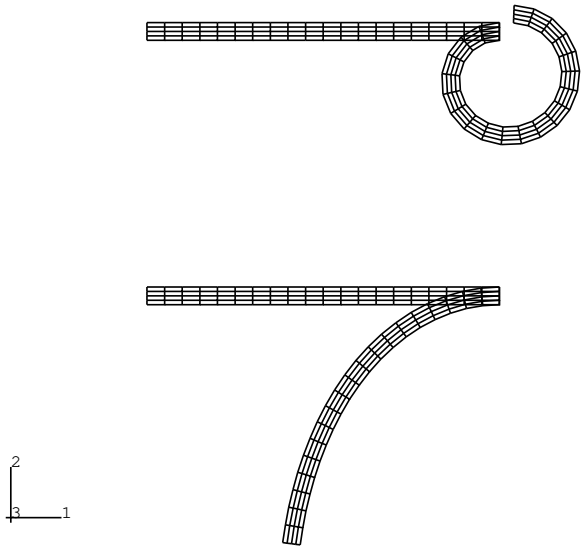


Figure 1: Deformed and undeformed beam meshes (CPE4R).

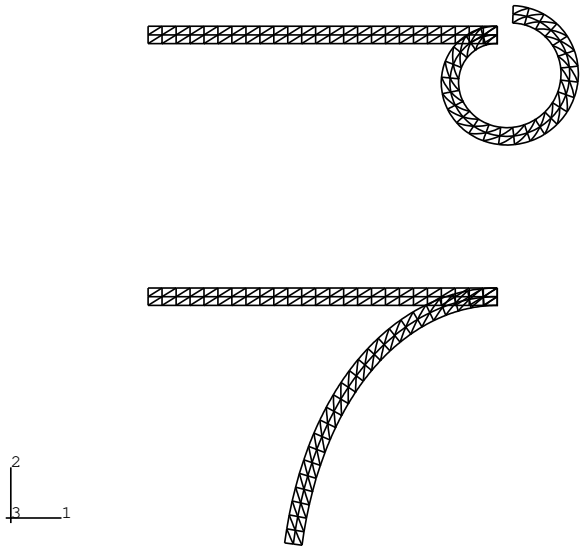


Figure 2: Deformed and undeformed beam meshes (CPE6M).

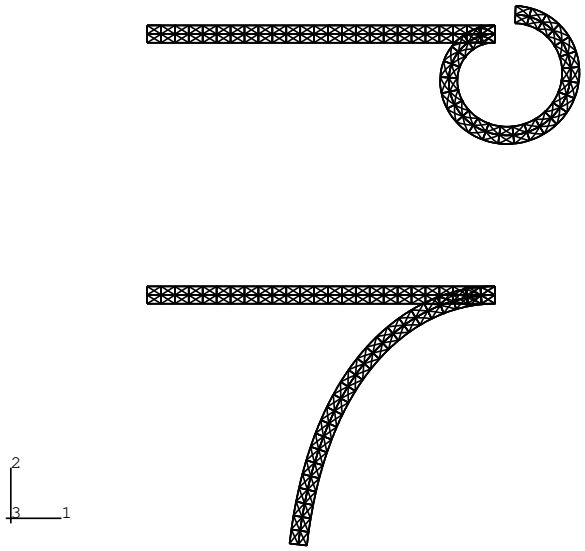


Figure 3: Deformed and undeformed beam meshes (C3D10M).

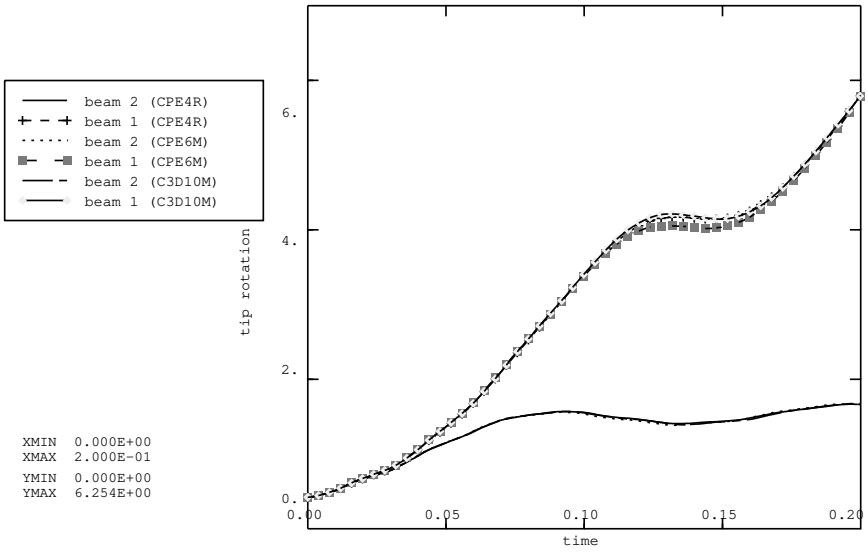


Figure 4: Tip rotations of beams.

Flexure of a deep beam

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

CPE4R C3D8R

Features tested

Hourglass control, kinematic formulation, tied contact surfaces, multi-point constraints.

Problem description

In this example the flexural response of a simply supported beam is modeled using continuum elements. The problem was originally used by *Flanagan and Belystchko (1982)* to test the hourglass control algorithms found in lower-order elements.

The half-symmetry model of the beam has a half-span of 0.4 and a depth of 0.1. The mesh consists of 32 elements (8×4). The material is linear elastic with Young's modulus = 1×10^6, Poisson's ratio = 0.0, and density = 1000.

A pinned boundary condition (directions 1 and 2) is specified for the center node on the left boundary of the mesh. A symmetry condition (direction 1) is specified for all the nodes on the right boundary of the mesh. A constant pressure load of magnitude 720000 is applied instantaneously to the top surface of the beam at the beginning of the step.

This problem is modeled with both two-dimensional and three-dimensional elements. In the two-dimensional case all the elements are 4-node plane strain continuum elements (CPE4R). *Figure 1* shows three meshes for the problem. The upper mesh is the standard case with 45 nodes. The center and lower mesh in the figure have been generated as two distinct parts each containing 16 elements (4×4) and 25 nodes. The two parts intersect along a vertical line of nodes where there are two nodes at each point with identical coordinates (coincident nodes). The mesh shown in the center is constrained to behave as the continuous mesh by using multi-point constraints to pin the coincident nodes along the interface between the two parts. In the lower mesh a surface-based tie constraint is used to constrain the nodes along the interface to have the same response as the original mesh. The three meshes should give identical results with these constraints. All the nodes that have boundary conditions or constraints are indicated in *Figure 1* by circles.

The three-dimensional case is identical to the two-dimensional case except that 8-node continuum elements (C3D8R) are used to model the beam. In this case the out-of-plane displacements are constrained to be zero (plane strain). Three meshes are also used in the three-dimensional case with the same constraints (in three dimensions) as described for the two-dimensional case.

The above problems are solved with different section control options. For two-dimensional and three-dimensional solid elements the section control options in Abaqus/Explicit allow the user to choose between five different hourglass control options. In addition, three different kinematic assumptions can be chosen for the three-dimensional solid elements. A discussion of the accuracy and performance that can be obtained with the various section control options can be found in *Section Controls*. Viscous hourglass control should not be used in quasi-static or low-mode dynamics problems, and analyses with this option are not included here. The section controls option in Abaqus/Standard allows the user to pick between two different hourglass control options. The

reduced-integration elements in Abaqus/Standard allow only average strain kinematic formulation with second-order accuracy. *Table 1* lists the various options and their plot legend and file descriptors.

Results and discussion

Figure 2 through *Figure 4* show results for the two-dimensional analysis run with default section control options (relaxed stiffness hourglass control is used) with Abaqus/Explicit. *Figure 2* shows the deformed shape for the two-dimensional case at the maximum deflection (time=.016). The three-dimensional deformed shapes are indistinguishable from those for the two-dimensional case. *Figure 3* shows the time history of vertical deflection for the midpoint on the symmetry plane for the two-dimensional case. There are three values plotted in the figure (one for each mesh), and they are identical. *Figure 4* shows the time history of the energies in the two-dimensional case. *Figure 5* and *Figure 6* show results for the three-dimensional analysis run with default section control options (average strain kinematic and relaxed stiffness hourglass control are used) with Abaqus/Explicit. *Figure 5* shows the time history of vertical deflection for the midpoint on the symmetry plane for the three-dimensional case. *Figure 6* shows the time history of the energies in the three-dimensional case. All three values (one for each mesh) are plotted. The results correspond exactly with the results reported in *Flanagan and Belytschko (1982)*.

For this problem only slight differences are observed among the default and nondefault kinematic and hourglass options in Abaqus/Explicit. With enhanced hourglass control, the solution for the two-dimensional case essentially matches the three-dimensional case with average strain kinematics. *Figure 7* through *Figure 9* show the history of the tip displacement for selected nondefault section control cases. *Table 1* lists the peak response of the vertical displacements for all of the cases.

The two-dimensional and three-dimensional analyses were also run in Abaqus/Standard with enhanced and stiffness hourglass control. *Figure 10* compares the time history of the tip displacement for enhanced hourglass control for the two-dimensional case between Abaqus/Standard and Abaqus/Explicit. The Abaqus/Explicit analysis was run with the average strain kinematic formulation and second-order accuracy, which are the only options available in Abaqus/Standard. The results show a close match. The results obtained using stiffness hourglass control and nondefault hourglass stiffness with Abaqus/Standard also agree with the results obtained with enhanced hourglass control for both the two-dimensional and three-dimensional analyses.

Input files

bend2d_cs.inp	Two-dimensional case with the COMBINED hourglass control.
bend2d_enhs.inp	Two-dimensional case with the ENHANCED hourglass control.
bend2d_rs.inp	Two-dimensional case with the default section control options (RELAX STIFFNESS hourglass control).
bend2d_ss.inp	Two-dimensional case with the STIFFNESS hourglass control.
bend2d_enhs_std.inp	Two-dimensional case with ENHANCED hourglass control in Abaqus/Standard.
bend2d_ss_std.inp	Two-dimensional case with STIFFNESS hourglass control in Abaqus/Standard.
bend3d_acs.inp	Three-dimensional case with the AVERAGE STRAIN kinematic and COMBINED hourglass control options.
bend3d_aenhs.inp	Three-dimensional case with the AVERAGE STRAIN kinematic and ENHANCED hourglass control options.
bend3d_ars.inp	Three-dimensional case with the default section control options (AVERAGE STRAIN kinematic and RELAX STIFFNESS hourglass control).
bend3d_ass.inp	Three-dimensional case with the AVERAGE STRAIN kinematic and STIFFNESS hourglass control options.
bend3d_ccs.inp	Three-dimensional case with the CENTROID kinematic and COMBINED hourglass control options.

bend3d_cenhs.inp	Three-dimensional case with the CENTROID kinematic and ENHANCED hourglass control options.
bend3d_crs.inp	Three-dimensional case with the CENTROID kinematic and RELAX STIFFNESS hourglass control options.
bend3d_css.inp	Three-dimensional case with the CENTROID kinematic and STIFFNESS hourglass control options.
bend3d_ocs.inp	Three-dimensional case with the ORTHOGONAL kinematic and COMBINED hourglass control options.
bend3d_oenhs.inp	Three-dimensional case with the ORTHOGONAL kinematic and ENHANCED hourglass control options.
bend3d_ors.inp	Three-dimensional case with the ORTHOGONAL kinematic and RELAX STIFFNESS hourglass control options.
bend3d_oss.inp	Three-dimensional case with the ORTHOGONAL kinematic and STIFFNESS hourglass control options.
bend3d_aenhs_std.inp	Three-dimensional case with ENHANCED hourglass control in Abaqus/Standard.
bend3d_ass_std.inp	Three-dimensional case with STIFFNESS hourglass control in Abaqus/Standard.

References

- Flanagan, D. P., and T. Belystchko, “A Uniform Strain Hexahedron and Quadrilateral with Orthogonal Hourglass Control,” J. Comp. Meth. Appl. Mech. Eng., vol. 17, pp. 679–706, 1982.

Tables

Table 1: Peak response of the vertical displacement of the centerline of the beam for different section control options.

Analysis File	Peak Response (× 10^{-2})	Section Controls	
		Kinematic	Hourglass
bend2d_rs	–6.638	n/a	relax
bend2d_ss	–6.630	n/a	stiffness
bend2d_cs	–6.743	n/a	combined
bend2d_enhs	–6.409	n/a	enhanced
bend2d_enhs_std	–6.394	n/a	enhanced
bend2d_ss_std	–6.423	n/a	stiffness
bend3d_ars	–6.466	average	relax
bend3d_ass	–6.451	average	stiffness
bend3d_acs	–6.566	average	combined
bend3d_aenhs	–6.401	average	enhanced
bend3d_ors	–6.466	orthogonal	relax
bend3d_oss	–6.451	orthogonal	stiffness
bend3d_ocs	–6.566	orthogonal	combined

Analysis File	Peak Response (× 10^{-2})	Section Controls	
		Kinematic	Hourglass
bend3d_oenhs	–6.401	orthogonal	enhanced
bend3d_crs	–6.464	centroid	relax
bend3d_css	–6.449	centroid	stiffness
bend3d_ccs	–6.565	centroid	combined
bend3d_cenhs	–6.392	centroid	enhanced
bend3d_aenhs_std	–6.394	n/a	enhanced
bend3d_ass_std	–6.286	n/a	stiffness

Figures

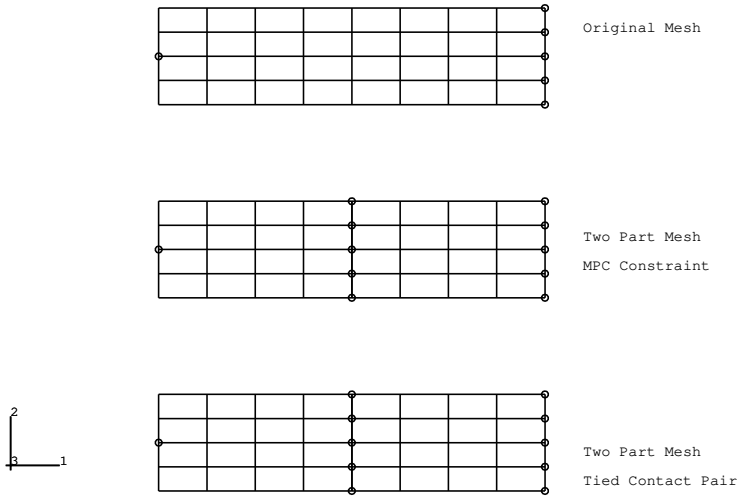


Figure 1: Original mesh for flexure.

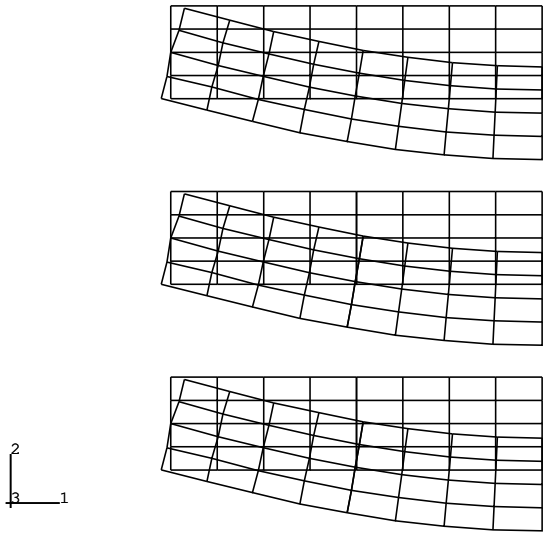


Figure 2: Deformed mesh at T=0.016 sec (2D case with default section controls).

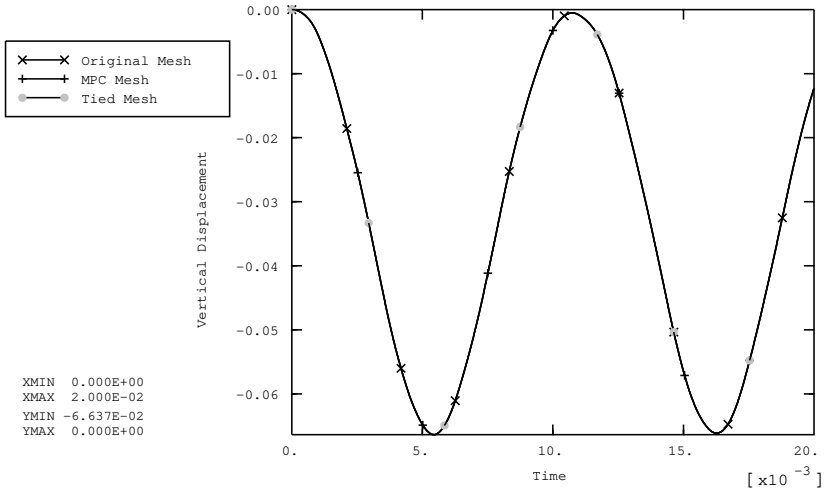


Figure 3: History of the vertical displacement of the centerline (2D case with default section controls).

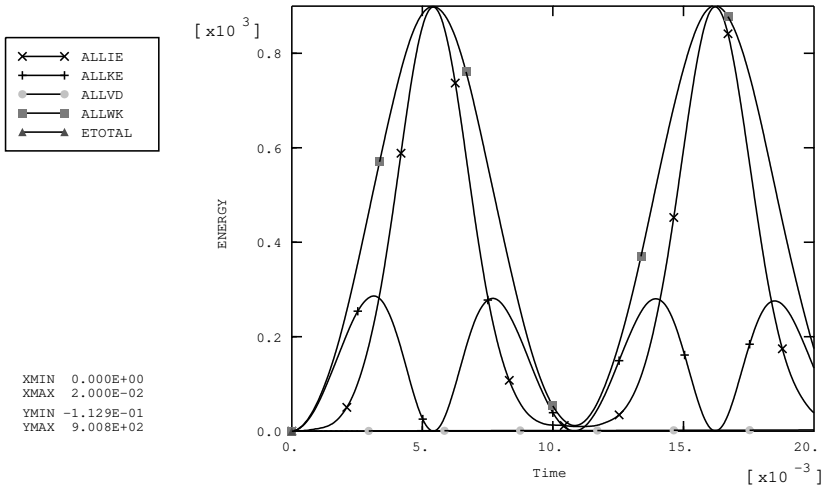


Figure 4: Time history of the energies (2D case with default section controls).

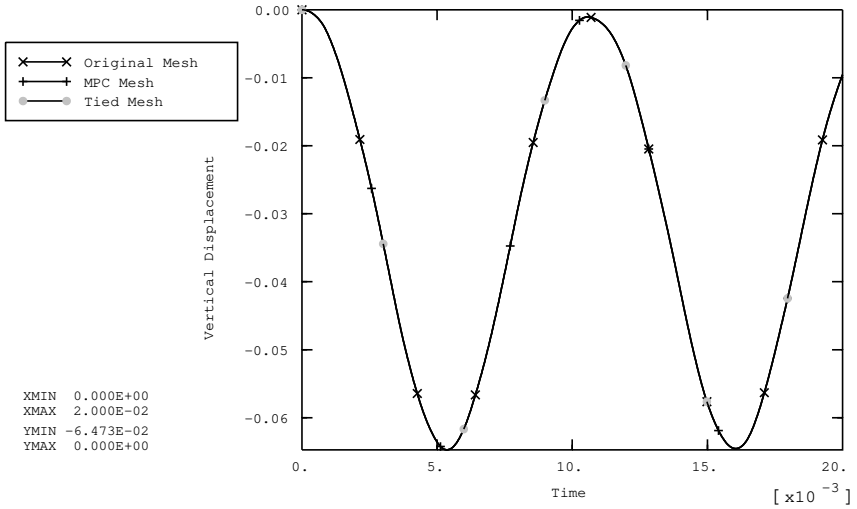


Figure 5: History of the vertical displacement of the centerline (3D case with default section controls).

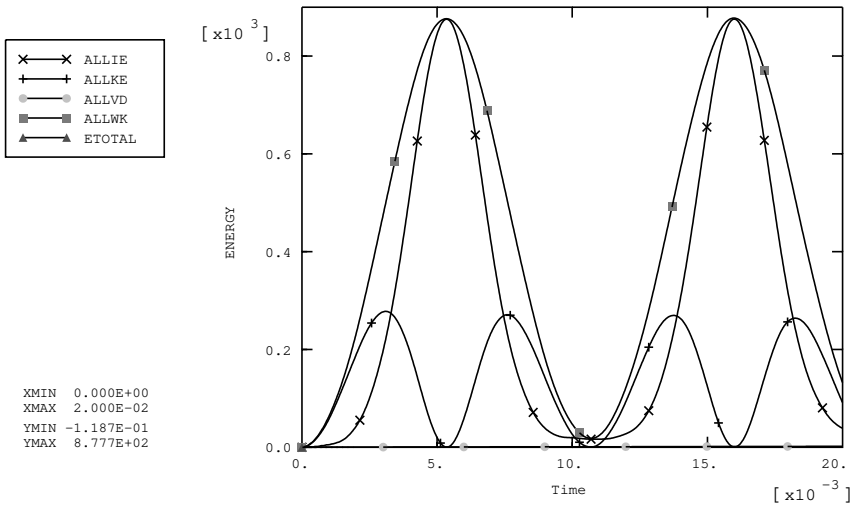


Figure 6: Time history of the energies (3D case with default section controls).

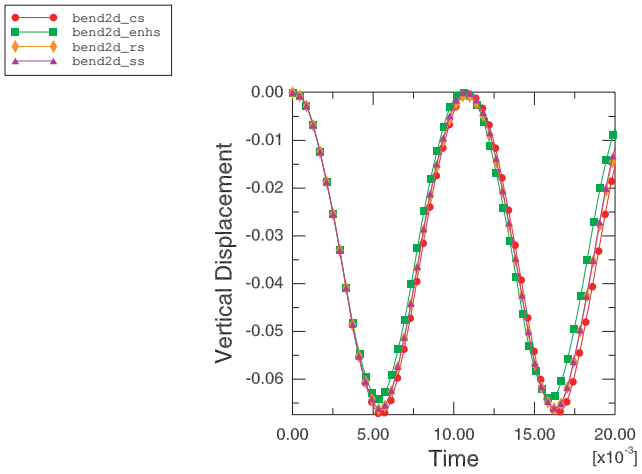


Figure 7: Comparison of the tip displacement history for the 2D case with different section control options (original mesh).

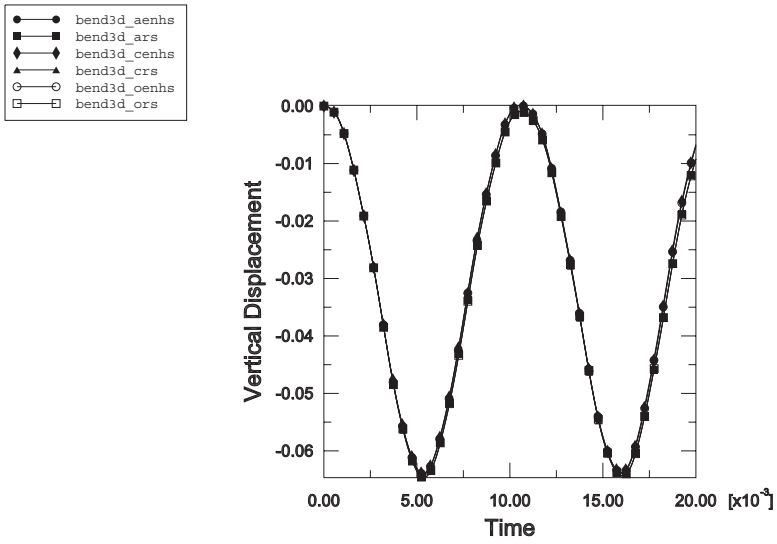


Figure 8: Comparison of the tip displacement history for the 3D case with different section control options (original mesh).

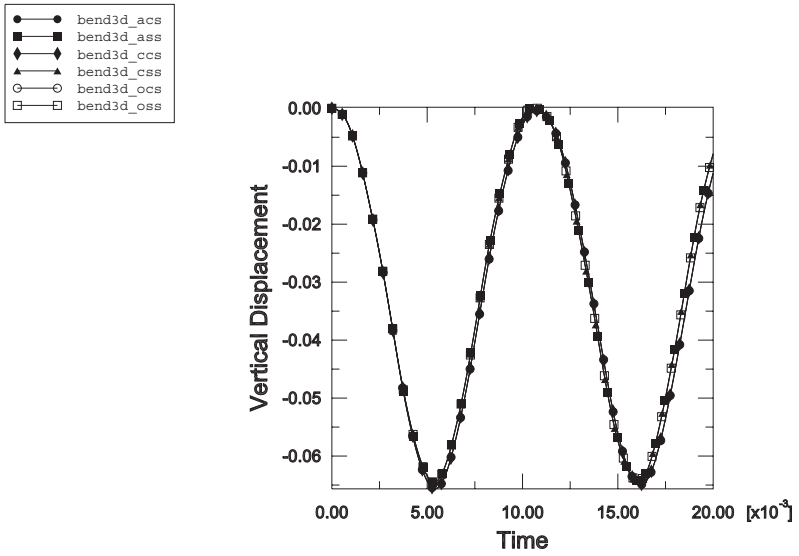


Figure 9: Comparison of the tip displacement history for the 3D case with different section control options (original mesh).

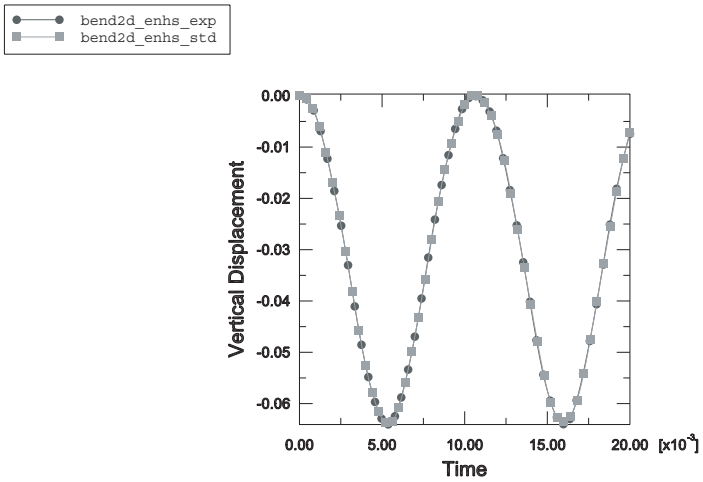


Figure 10: Comparison of the tip displacement history for the 2D case with enhanced hourglass control (original mesh).

Simple tests of beam kinematics

Products: Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

B21 B22 B31 B32 PIPE21 PIPE31

Features tested

Stability of beams in deformation and in rigid body rotation.

Problem description

This problem is used to verify that individual beam elements demonstrate stable behavior for both small-displacement response and large-rotation response. In the first case the beam is loaded in the axial, bending, shear, and twisting (three-dimensional beams only) deformation modes and allowed to vibrate freely. The second case tests rigid body rotation of a beam about one of its endpoints. In both cases two-dimensional and three-dimensional beams are tested with and without bulk viscosity. Two-dimensional and three-dimensional pipe elements are also tested for deformations, similar to beam elements with pipe cross-sections.

Deformation tests

These tests consist of three steps. In the first step the bulk viscosity of the beam is set to zero, and a displacement or rotation is applied to the ends of the beam using a smooth step amplitude. In the second step the displacement constraints are removed, and the beam is allowed to oscillate freely. Finally, in the third step the bulk viscosity is set to a value of 0.06 and the beam is allowed to oscillate with damping. Fixed time incrementation is used in all of the steps. This time incrementation strategy uses a time increment that is based on the critical element-by-element stable time increment estimates at the beginning of a step. It is used to avoid the propagation of noise in the solution that may occur when the default time incrementation strategy is used without bulk viscosity. Normally, the default bulk viscosity will damp out and prevent the propagation of this high-frequency noise.

Rigid body rotation tests

These tests consist of two steps. Initial velocities are applied to the beam to induce rotation, and initial axial stresses are applied to simulate the centrifugal stress generated in a rotating body. In the first step the bulk viscosity is set to zero and the beam is allowed to rotate 5 complete revolutions about its endpoint. In the second step the bulk viscosity is set to 0.06 and the beam is allowed to rotate another 5 revolutions. In the two-dimensional case the axis of rotation is the z-axis. In the three-dimensional case the axis of rotation is in the X–Y plane aligned at -45° to the original y-axis.

Results and discussion

The results for each test are described in the following sections.

Deformation test results

This problem demonstrates that the beam elements used in Abaqus/Explicit provide stable behavior for free and damped vibration. *Figure 1*, *Figure 2*, and *Figure 3*, respectively, show typical displacement and rotation results for the axial, bending, and shear loading of a two-dimensional beam with a box cross-section. All displacements and rotations exhibit magnitudes equal to or less than those applied in Step 1.

The energy balance for the axially loaded beam is poor, as shown in *Figure 4*. This inaccuracy occurs because too few increments are used to predict each cycle of the beam's axial response. The inaccuracy occurs only in the axially loaded case because the period of the vibration in the other modes is significantly higher, so more time increments are included in each vibration cycle. The displacement response and energy balance can be obtained more accurately by using direct time integration. The results obtained for the axial response of the two-dimensional box-section beam using direct time integration with a time increment of 1×10^{-4} are shown in *Figure 5* (displacement) and *Figure 6* (energy balance).

Rotation test results

All axial strains are zero for both the two-dimensional and the three-dimensional cases. The displacement in the *x*-direction varies sinusoidally with a constant amplitude over the entire range of rotation. Plots of the displacement in the *x*-direction versus time are shown in *Figure 7* for the two-dimensional case and *Figure 8* for the three-dimensional case.

Input files

The input files included with the Abaqus release are named according to the following convention: b*dimension_x-section_loading*.inp, where

dimension

indicates the dimension. The keys are 2d for two-dimensional beams, and 3d for three-dimensional beams.

x-section

indicates the cross-section of the beam used in the analysis. The keys are:

box

for a box cross-section,

circ

for a circular cross-section,

hex

for a hexagonal cross-section,

i

for an I-section,

l

for an L-section,

pipe

for a pipe section,

rect

for a rectangular section,

trap

for a trapezoidal section,

arb_o

for an arbitrary open section, and

arb_c

for an arbitrary closed section.

gs

for beam general section with linear response.

gsbox

for beam general section with a rectangular, hollow box section.

gsl

for beam general section with nonlinear response on the cross-section and linear axial, first bending moment, second bending moment, and torque response.

gsnl

for beam general section with nonlinear response on the cross-section and elastic axial, first bending moment, second bending moment, and torque response.

gsp

for beam general section with nonlinear response on the cross-section and axial, first bending moment, second bending moment, and torque response.

loading

indicates the displacement mode for the analysis. The loading keys are:

axial

for axial deformation,

bend

for bending,

shear

for shear deformation,

twist

for twisting deformation about the beam axis (three-dimensional beams only), and

rot

for rigid body rotation of the beam (circular cross-section only).

Additional input files for analyses of the box, circular, L, and rectangular cross-sections with an effective Poisson's ratio include an _p after the loading parameter. For example:

b2d_box_axial.inp	Two-dimensional beam element with a box cross-section and axial loading.
b3d_circ_twist_p.inp	Three-dimensional beam element with a circular cross-section, applied twist, and an effective Poisson's ratio defined for the section.

Figures

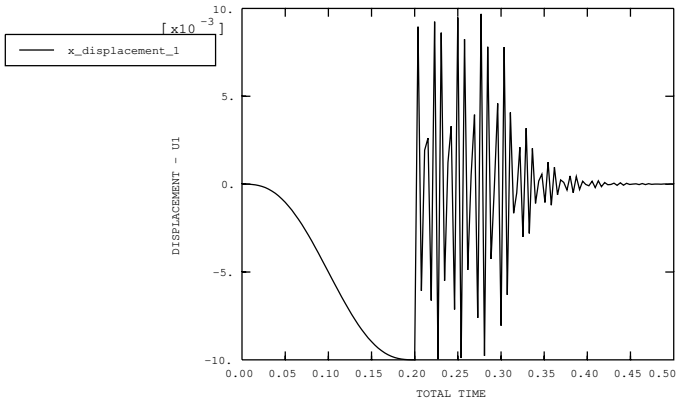


Figure 1: B21 box cross-section with axial displacements.

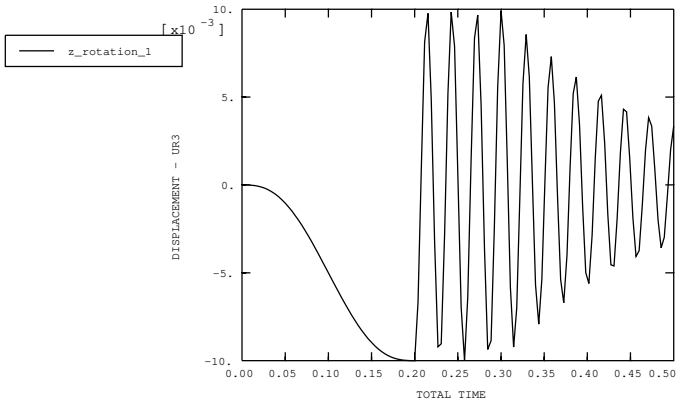


Figure 2: B21 box cross-section with bending.

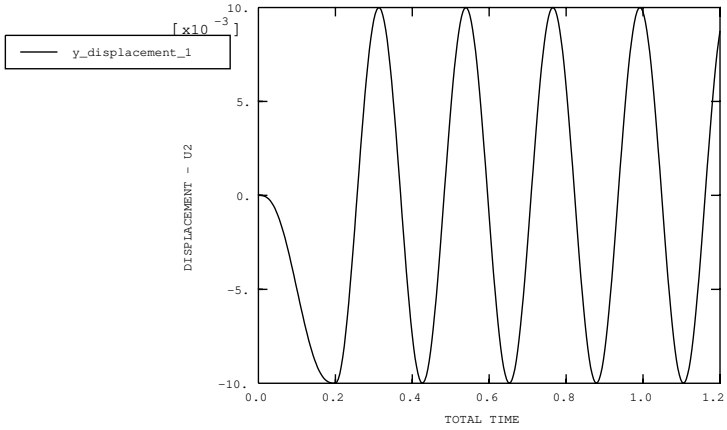


Figure 3: B21 box cross-section with shearing displacements.

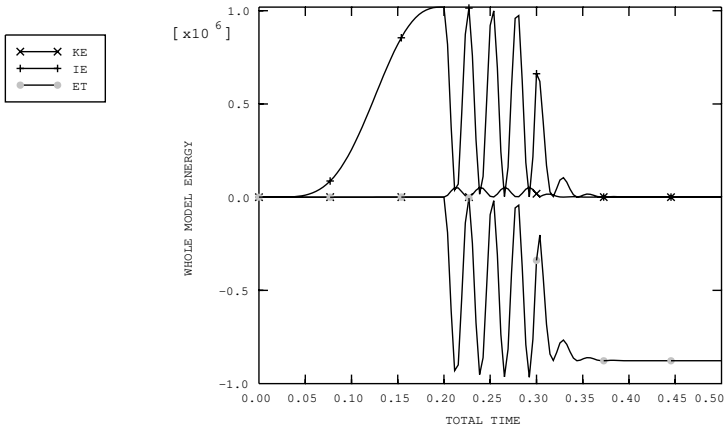


Figure 4: Energies for axial displacements (FIXED time increment control).

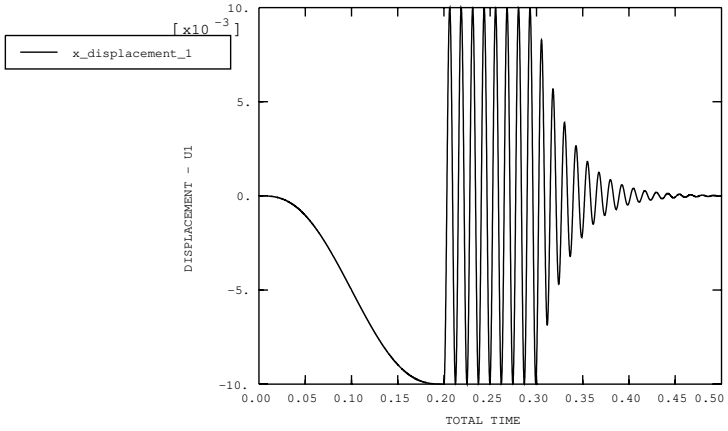


Figure 5: B21 box-section with axial displacements (direct-solution time increment control).

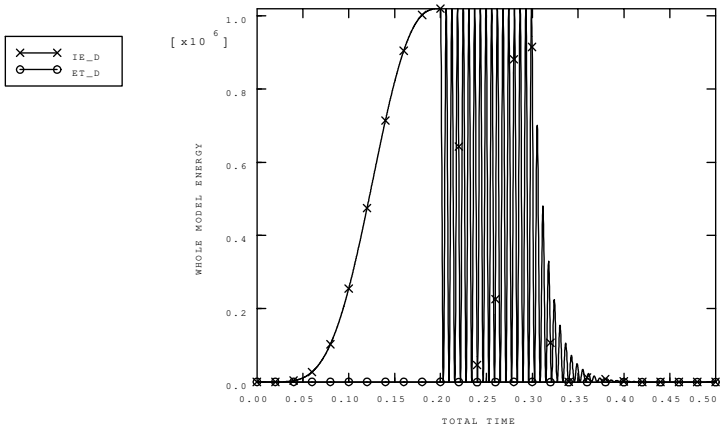


Figure 6: Energies for axial displacements (direct-solution time increment control).

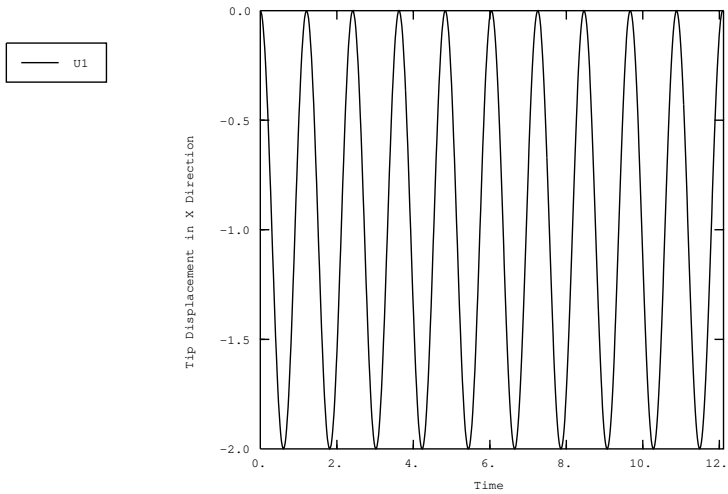


Figure 7: B21 displacement in x-direction versus time.

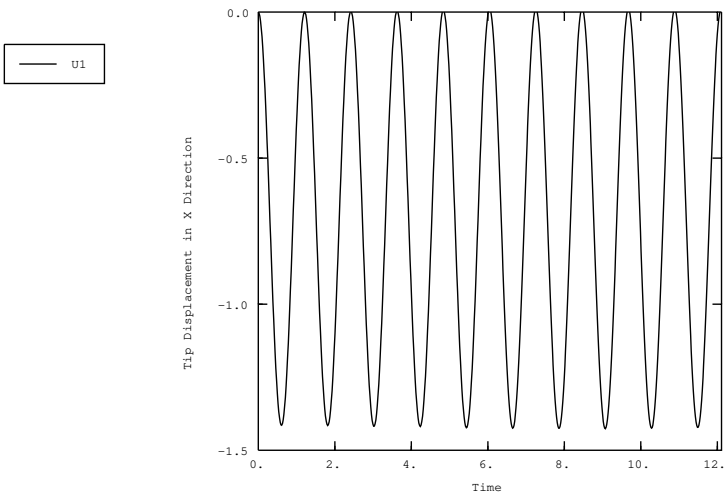


Figure 8: B31 displacement in x-direction versus time.

Tensile test

Products: Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

CPS3 CPE3 CAX3 C3D6 M3D3 S3R C3D4

Features tested

Concentrated loads.

Problem description

Elements are subjected to tensile loading in this problem. The problem is analyzed using seven different element types. The mesh is shown in *Figure 1*.

The material model is isotropic linear elasticity. The material properties used are Young's modulus = 1.0, Poisson's ratio = 0.0, and density = 1. Taking advantage of the symmetry of the configuration, the bottom of the model in each case is constrained against displacement in the vertical direction, and the left side is constrained against displacement in the horizontal direction.

The magnitude of the concentrated load is chosen such that the total strain is .01. The load magnitude is increased linearly from zero to its final value over the first half of the step; it is then held constant over the second half of the step to verify that any oscillatory dynamic effects are minimal.

Results and discussion

Figure 2 shows the elements in their displaced configuration, with the displacements magnified by a factor of 50. *Figure 3* shows a history plot of vertical displacement versus time for each of the seven cases. Since Poisson's ratio is 0.0, the results for the seven cases are identical.

Input files

tensile.inp	Input data used in this analysis.
tensile_c3d4.inp	C3D4 elements.
tensile_c3d6.inp	C3D6 elements.
tensile_c3d8r.inp	C3D8R elements.
tensile_cax3.inp	CAX3 elements.
tensile_cax4r.inp	CAX4R elements.
tensile_cpe3.inp	CPE3 elements.
tensile_cpe4r.inp	CPE4R elements.
tensile_cps3.inp	CPS3 elements.
tensile_cps4r.inp	CPS4R elements.
tensile_m3d3.inp	M3D3 elements.

tensile_m3d4r.inp	M3D4R elements.
tensile_s3r.inp	S3R elements.
tensile_s4r.inp	S4R elements.
tensile_s3r_gauss2.inp	Shell elements with Gauss integration, 2 Gauss integration points used for the shell section integration.
tensile_s3r_gauss4.inp	Shell elements with Gauss integration, 4 Gauss integration points used for the shell section integration.
tensile_s3r_gauss5.inp	Shell elements with Gauss integration, 5 Gauss integration points used for the shell section integration.
tensile_s3r_gauss6.inp	Shell elements with Gauss integration, 6 Gauss integration points used for the shell section integration.
tensile_s3r_gauss7.inp	Shell elements with Gauss integration, 7 Gauss integration points used for the shell section integration.

Figures

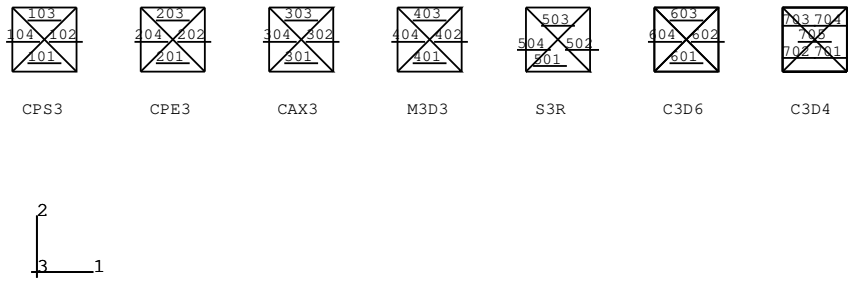


Figure 1: Mesh for tensile test problem.

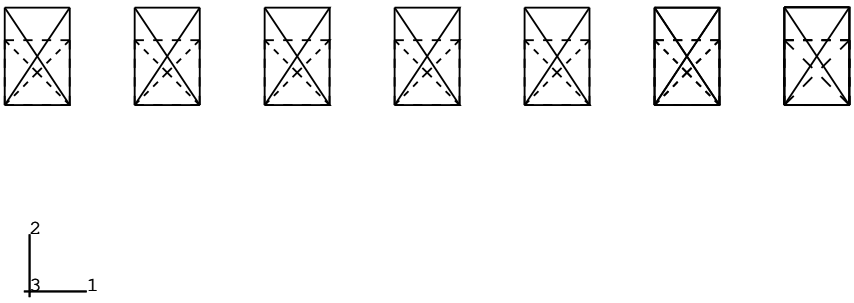


Figure 2: Displaced elements in tensile test problem.

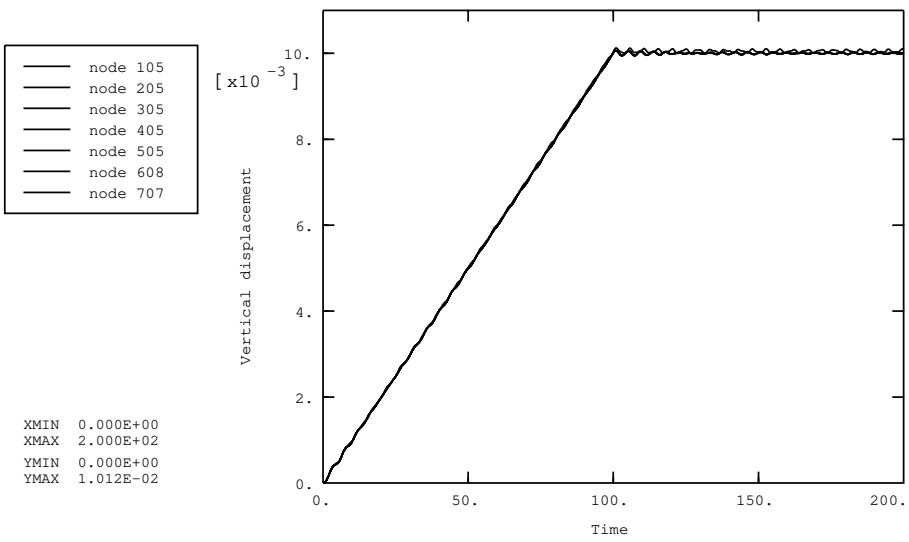


Figure 3: Vertical displacement versus time.

Simple shear

Products: Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

CPE4R CPS4R C3D8 C3D8R M3D4R S4 S4R

Features tested

Large deformation kinematics, stability of large-strain formulation.

Problem description

In this problem a state of simple shear is induced in a single element up to a nominal shear strain of 300%. The material model is isotropic linear elasticity. There are no physical materials that exhibit linear elastic response to such large shear strain. The purpose of this example problem is to verify the large deformation and large rotation algorithms in Abaqus/Explicit.

The material properties used are Young's modulus = 1.0, Poisson's ratio = 0.0, and density = 1.346×10^{-4}.

In this problem all the in-plane degrees of freedom are either zero or are prescribed as functions of time. The value used for the density controls the time increment size, and it was chosen to give a time increment size that results in about 1% shear strain per increment.

This problem is analyzed using five different element types, each of which is defined twice. Each element in the bottom row is sheared in the x-direction; each element in the top row is sheared in the y-direction.

Results and discussion

The computed stress-strain curves for the bottom and top rows of elements are in agreement with analytic solutions.

These results demonstrate that the kinematic formulation is uniform across all the element types defined in Abaqus/Explicit.

Input files

shear.inp	C3D8R, CPE4R, CPS4R, M3D4R, and S4R elements.
shear_c3d4.inp	C3D4 element.
shear_c3d6.inp	C3D6 element.
shear_c3d8r.inp	C3D8R element.
shear_cax3.inp	CAX3 element.
shear_cpe3.inp	CPE3 element.
shear_cpe4r.inp	CPE4R element.
shear_cps3.inp	CPS3 element.

shear_cps4r.inp	CPS4R element.
shear_m3d3.inp	M3D3 element.
shear_m3d4r.inp	M3D4R element.
shear_s3r.inp	S3R element.
shear_s4r.inp	S4R element.
shear_c3d8.inp	C3D8 element included for the purpose of testing performance only.
shear_s4.inp	S4 element included for the purpose of testing performance only.

Verification of the elastic behavior of frame elements

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Simple load tests

Elements tested

FRAME2D FRAME3D

Features tested

The elastic behavior of frame elements is tested. The different cross-sections considered for the frame elements are:

- rectangular hollow box section
- solid circular section
- general cross-section
- I-beam section
- hollow circular section
- solid rectangular section

The tests are performed for the following load conditions:

- concentrated loads
- gravity loads
- force per unit length in the global and the local frame directions
- thermal loads as nodal temperatures
- transverse fluid drag load
- fluid drag force on the ends of the frame
- tangential fluid drag load
- buoyancy load (closed-end condition)
- transverse wind drag load
- wind drag force on the ends of the frame
- foundation loads in the global and local frame directions

These loads are considered to act either individually or in combination. Both regular static steps and linear perturbation steps are considered.

The local coordinate system is also tested. Temperature dependence of frame element properties is tested under thermal loading. The initial stress conditions and the initial temperature at the nodes are also verified.

Problem description

The problem consists of a cantilever with a length of 75.0 units made of five frame elements. Various orientations of the cantilever in space are considered. The cross-sectional dimensions shown in *Verification of beam elements and section types* are used for the five section types (rectangular hollow box, solid circular, hollow circular, rectangular, and I-beam).

The cantilever is subjected to concentrated tip loading that leads to both flexure and torsion. The wind loads, WD1 and WD2, and the Aqua loads, FD1 and FD2, also apply concentrated forces at the nodes. The remaining loads cause uniformly distributed loading on the cantilever. Under thermal loading the free end of the cantilever is fixed. The wind velocity profile is made nearly uniform with the height by setting the exponent α to 1×10.0^{-9}. The fluid velocity in the Aqua loading is constant with height. With foundation loads the boundary conditions of the cantilever are changed to simple supports, and the cantilever is pressed uniformly into the foundation using distributed loads.

Material:

Young's modulus at temperature −10.0 units:	3×10^6
Poisson's ratio at temperature −10.0 units:	0.3
Young's modulus at temperature 90.0 units:	1.5×10^6
Poisson's ratio at temperature 90.0 units:	0.3
Reference temperature for definition of thermal expansion coefficient:	−10.0
Thermal expansion coefficient at −10.0 temperature:	0.001
Thermal expansion coefficient at 90.0 temperature:	0.002
Initial temperature:	−10.0
Material density:	0.8
Gravitational constant:	10.0
Density of air for wind loads:	0.008
Density of fluid for Aqua loads:	0.008
Seabed level:	−100.0
Still fluid level:	50.0
Foundation stiffness:	1500.0

Results and discussion

The problem is statically determinate. The section forces and section strains match the analytical values.

Input files

frame2d_bs_thermal.inp	Box section with thermal loading.
frame2d_cs_wind_transform.inp	Circular section with wind loading and **TRANSFORM*.
frame2d_gs_foundation.inp	General section with **FOUNDATION* loading.
frame2d_gs_sig0.inp	General section with initial stress, perturbation step with **LOAD CASE*.
frame2d_is_aqua.inp	I-section with Aqua fluid loading.
frame2d_ps_sig0.inp	Pipe section with initial stress.
frame2d_rs_aqua.inp	Rectangular section with Aqua fluid loading.

frame2d_rs_aqua_transform.inp	Rectangular section with Aqua fluid loading and **TRANSFORM*.
frame2d_rs_foundation.inp	Rectangular section with **FOUNDATION* loading.
frame3d_bs_wind.inp	Box section with wind loading.
frame3d_cs_foundation.inp	Circular section with **FOUNDATION* loading.
frame3d_cs_transform.inp	Circular section with **TRANSFORM*.
frame3d_gs_sig0_transform.inp	General section with initial stress.
frame3d_is_aqua.inp	I-section with Aqua fluid loading.
frame3d_ps_foundation.inp	Pipe section with **FOUNDATION* loading.
frame3d_ps_thermal.inp	Pipe section with thermal loading.
frame3d_rs_sig0_transform.inp	Rectangular section with initial stress and **TRANSFORM*.

Elastic frame elements with pinned ends

Elements tested

FRAME2D FRAME3D

Features tested

The linear elastic uniaxial behavior of frame elements under a concentrated load is tested.

Problem description

Pinned connections are specified for the ends of frame elements by declaring the relevant parameter for the frame section. In this example the frame element behaves as an axial spring with constant stiffness. In small-displacement analysis the element can be compared with truss or spring elements. The model and geometry used are the same as in the verification problem *Three-bar truss*.

Results and discussion

All tests match the exact solution; for details, see *Three-bar truss*.

Input files

frame2d_3bar_pinned.inp	Rectangular section with **CLOAD* loading.
frame3d_3bar_pinned.inp	Rectangular section with **CLOAD* loading.

Elastic frame elements with buckling strut response

Elements tested

FRAME2D FRAME3D

Features tested

The uniaxial buckling strut behavior of frame elements with both ends pinned is tested.

Problem description

The buckling strut envelope corresponds to Marshall Strut theory. The tests consist of one frame element fixed at one end and subjected to a prescribed displacement on the other. The value of the prescribed displacement changes according to an amplitude definition. The variation of the amplitude is chosen in such a way that the

buckling strut envelope is traced for the compressive as well as for the tensile behavior up to and beyond the yield stress value. Uniaxial response, buckling strut response, and yield stress are considered for the cross-section for frame elements.

Model:

Pipe's radius:	2.
Pipe's thickness:	0.08122693
Cross-sectional area:	1.

Material:

Young's modulus:	30×10^6
Shear modulus:	10×10^6
Yield stress:	1×10^6

Results and discussion

The uniaxial buckling and postbuckling behavior in compression and isotropic hardening behavior in tension can be seen by plotting the axial force in the element against the prescribed displacement; see *Figure 1*.

Input files

frame2d_pinned_buckl.inp	Pipe section with prescribed displacement.
frame3d_pinned_buckl.inp	Pipe section with prescribed displacement.

Figures

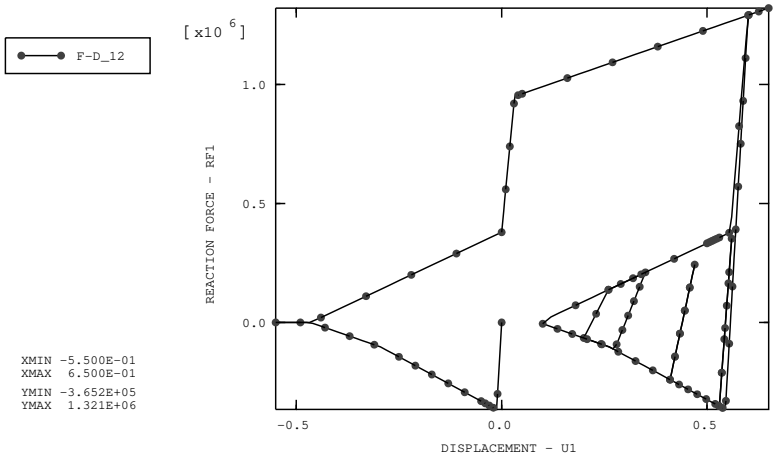


Figure 1: Buckling response of FRAME2D element.

Elastic frame element with buckling strut response for nonlinear geometry

Elements tested

FRAME2D

Features tested

A collapsing scaffold is investigated in a geometrically nonlinear analysis.

Problem description

The scaffold is made of three pinned frame elements with pipe cross-sections. The buckling strut envelope corresponds to Marshall Strut theory. The collapse occurs under a force-controlled loading.

Model:

Pipe's radius:	0.2
Pipe's thickness:	0.01

Material:

Young's modulus:	$3. \times 10^6$
Shear modulus:	1.5×10^6
Yield stress:	51.9×10^3

Results and discussion

The snap-through character of the response requires the Riks analysis procedure. *Figure 2* plots the section force in each element versus the load factor from the Riks analysis. The buckling of frame elements 2 and 3 changes the force distribution of the entire structure. After element 3 buckles, it remains buckled throughout the loading process; element 2 buckles, then regains stiffness and develops tensile force, as seen in *Figure 2*.

Input files

frame2d_pinned_buckl_nlgeom.inp	Buckling pipe section with nonlinear geometry.

Figures

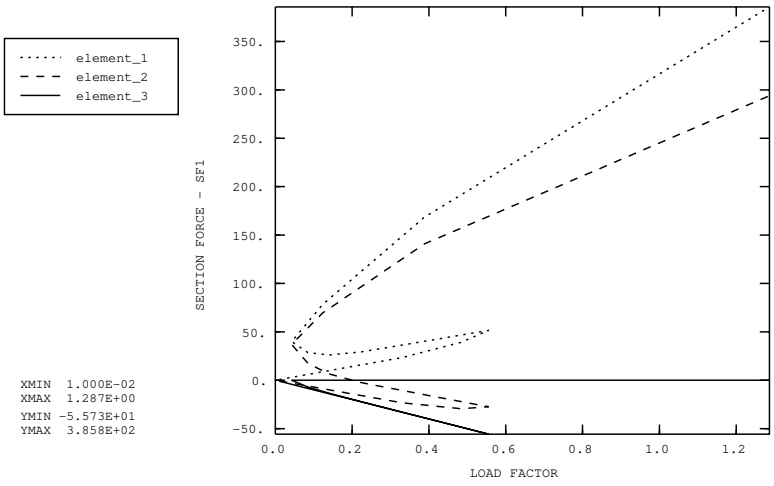


Figure 2: Buckling response of scaffold with FRAME2D pipe elements.

Elastic frame elements with switching algorithm for nonlinear geometry

Elements tested

FRAME2D FRAME3D

Features tested

A collapsing scaffold with geometry and material properties as described in *Elastic frame element with buckling strut response for nonlinear geometry* is investigated using frame elements with the switching algorithm.

Problem description

The buckling strut response is enabled for the elastic frame elements. The ISO equation is used as a criterion for the switching algorithm, and the default buckling envelope governs the postbuckling behavior.

Results and discussion

Two types of problems are tested here: an in-plane scaffold structure modeled with FRAME2D and FRAME3D elements and a three-dimensional scaffold supported by an additional out-of-plane element. The default buckling envelope is used for the in-plane scaffold problems, and a nondefault buckling envelope is used in the three-dimensional scaffold. In all problems the buckling reduction factors are 1.0 in both directions. All end points of the scaffold structure are fixed, and a prescribed displacement is applied to node 2. The value of the displacement is chosen such that elements 1, 3, and 4 in the three-dimensional scaffold will violate the ISO equation and, therefore, will cause a switch to strut response.

Figure 3 plots the axial force in elements 1 and 3 versus the time for the scaffold in plane. Element 3 buckles at the value of critical compressive force $P_{cr} = -56.75$ and loses its stiffness at 58% of the prescribed displacement values; element 1 buckles next and retains a small stiffness through the loading history.

The behavior of the three-dimensional scaffold is different. The first element that switches to the strut response is element 4, followed by elements 3 and 1. At 72.5% of the prescribed displacement values, elements 3 and 4 have already lost their stiffness.

Input files

frame2d_el_switch.inp	FRAME2D element with switching algorithm.
frame3d_el_switch.inp	FRAME3D element with switching algorithm.
frame3d_inspace_switch.inp	FRAME3D element with switching algorithm.

Figures

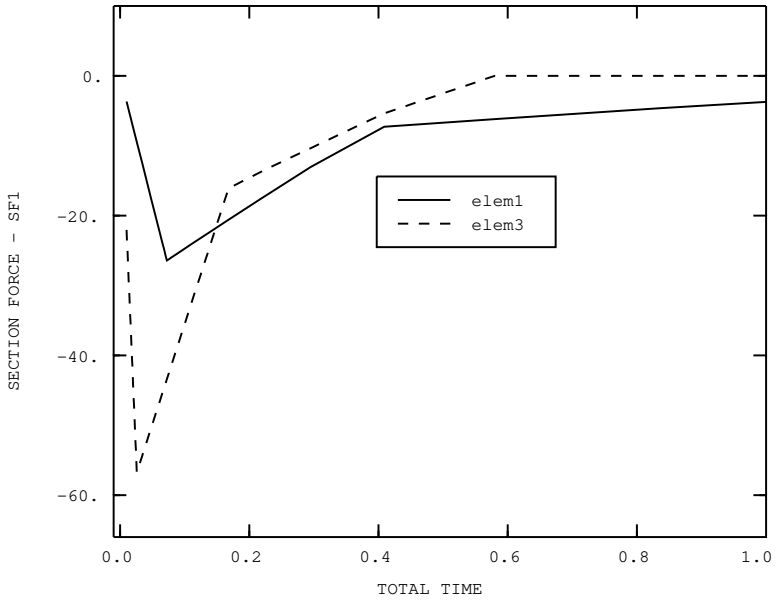


Figure 3: Buckling response of scaffold with FRAME2D and switching algorithm.

Verification of the plastic behavior of frame elements

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

FRAME2D FRAME3D

Features tested

The plastic behavior of frame elements with hollow circular, rectangular hollow box, and I-beam sections is tested under concentrated distributed loads. The yield surface is represented by an interaction of plastic axial forces with plastic moments including plastic torque. User-defined as well as default generalized plastic forces are used. Different geometries in two- and three-dimensional problems are considered.

Problem description

The first problem (*frame2d_pps_cload.inp*) consists of three plane frame elements with PIPE cross-sections forming a statically determinate system. In three load steps concentrated forces are applied at the nodes. The values for plastic axial force and plastic bending moment are user-defined.

In the second statically determinate system (*frame2d_pbs_cload.inp*), two frame elements are simply supported at both sides with concentrated forces applied at the middle node in the first load step. In the second load step an additional constant bending moment is applied to the system. The values for plastic axial force and plastic bending moment are user-defined.

The third example (*frame3d_pis_cload.inp*) is a one-element test in which an axial force, a bending load, and a torque are applied in three subsequent load steps. The plastic behavior is defined by default values from a given yield stress.

The fourth problem (*frame3d_pps_dload.inp*) is a statically determinate frame consisting of three elements that is loaded with various distributed loads, causing axial force, bending, and torque. The values for plastic axial force, plastic bending moment, and plastic torque are user-defined.

Model: Cross-sectional dimensions are given in the order required by the beam cross-sectional library.

Hollow circular cross-section: r, t	1., .174355
Rectangular hollow box cross-section: a, b, t_1, t_2, t_3, t_4	1., 1., 0.2, 0.2, 0.2, 0.2
I-beam cross-section: $l, h, b_1, b_2, t_1, t_2, t_3$	.2, .4, .1, .1, .015, .015, .01

Material:

Young's modulus:	3.0×10^6
Poisson's ratio:	0.3
Yield stress:	$50. \times 10^3$

Results and discussion

In all problems the plastic hinges were created at predicted locations indicated by the active yield flag. The value of the plastic displacement is given by requesting output variable SEP.

Input files

frame2d_pps_cload.inp	Plastic pipe section with concentrated loads.
frame2d_pbs_cload.inp	Plastic box section with concentrated loads, perturbation step with **LOAD CASE*.
frame3d_pis_cload.inp	Plastic I-section with concentrated load.
frame3d_pps_dload.inp	Plastic pipe section with distributed loads.

Three-bar truss

Products: Abaqus/Standard

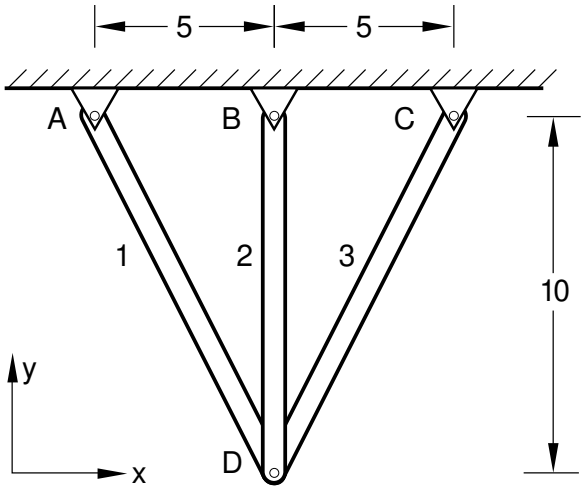
Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

T2D2 T2D2H T2D3 T2D3H T3D2 T3D2H T3D3 T3D3H
FRAME2D FRAME3D

Problem description



Material: Linear elastic, Young's modulus = 30.0×10^6.

Boundary conditions: Nodes *A*, ***B***, and *C* are pinned.

Loading: $F_y^D = -10000.0$.

Reference solution

$u_y^D = -1.3711 \times 10^{-2}$, $\sigma =$ 32907 in elements 1 and 3, $\sigma =$ 41134 in element 2.

Results and discussion

All elements yield exact solutions. Multi-point constraints are required to eliminate singularities in the three-node element tests using truss elements; e.g., T3D3.

The frame elements tested have rectangular cross-sections with the same cross-sectional area as the truss elements tested. In the cross-section for frame elements, pinned connections are specified for the ends of the frame elements by declaring the relevant parameter for the frame section. The joints are thus characterized by pinned connections. Since the frame elements are formulated in terms of section properties, stress output is not available; instead, the section forces are available. Stresses calculated from the axial force and the cross-sectional area match the stresses obtained from the truss element tests.

Input files

et22sfse.inp	T2D2 elements.
et22shse.inp	T2D2H elements.
et23sfse.inp	T2D3 elements.
et23shse.inp	T2D3H elements.
et32sfse.inp	T3D2 elements.
et32shse.inp	T3D2H elements.
et33sfse.inp	T3D3 elements.
et33shse.inp	T3D3H elements.
frame2d_3bar_pinned.inp	FRAME2D elements.
frame3d_3bar_pinned.inp	FRAME3D elements.

Pure bending of a cylinder: CAXA elements

Products: Abaqus/Standard

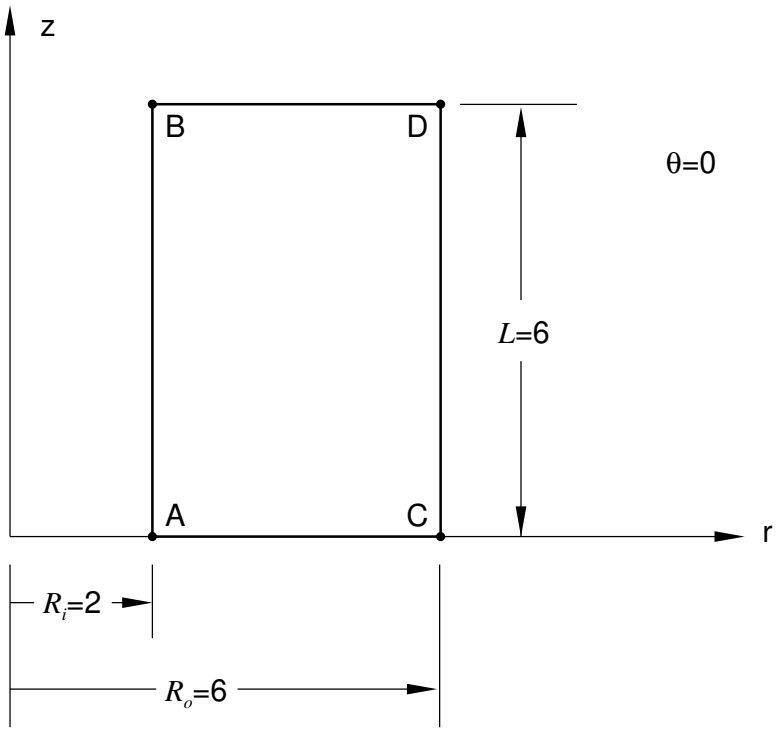
Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

CAXA4n CAXA4Rn CAXA8n CAXA8Rn (n =1, 2, 3, 4)

Problem description



A hollow cylinder of circular cross-section, inner radius R_i, outer radius R_o, and length $2L$ is subjected to a bending moment, M, applied to its end planes. For a linear elastic material with Young's modulus E and Poisson's ratio ν, the solutions for stress and displacement are as follows:

$$\begin{aligned}
\sigma_{zz} &= \frac{M}{I} r\cos\theta \\
\sigma_{rr} &= \sigma_{\theta\theta} = \sigma_{rz} = \sigma_{r\theta} = \sigma_{z\theta} = 0 \\
u_r &= \frac{-M}{2EI}\left[z^2 + \nu\left(r^2 - R_i^2\right)\right]\cos\theta \\
u_\theta &= \frac{M}{2EI}\left[z^2 - \nu\left(r^2 - R_i^2\right)\right]\sin\theta \\
u_z &= \frac{M}{EI} rz\cos\theta,
\end{aligned}$$

where $I = \pi\left(R_o^4 - R_i^4\right)/4$ is the moment of inertia of the cylinder and r, θ, and z are the cylindrical coordinates.

Only one-half of the structure is considered, with a symmetry plane at $z = 0$. The form of the displacement solution, which is a quadratic function in both r and z, suggests that a single second-order element should model the structure accurately. The full- and reduced-integration second-order elements do use a single element mesh, but an 8 × 12 mesh is used for the fully integrated first-order elements and a 16 × 24 mesh is used for the reduced-integration first-order elements.

Material: Linear elastic, Young's modulus = 30×10^6, Poisson's ratio = 0.33.

Boundary conditions: $u_z = 0$ on the $z = 0$ plane; at $r = R_i$ on the $z = 0$ plane, u_r at $\theta = 0°$ is set equal to u_r at $\theta = 180°$ with an equation constraint to remove the rigid body motion in the global x-direction.

Loading: The bending load is simulated by applying a surface traction of the form $\sigma_{zz} = \frac{M}{I} r\cos\theta$ on the $z = L$ plane of the cylinder. This is done by applying the appropriate nonuniform pressure load with a distributed load and defining the variation of the pressure in both the r- and θ-directions with user subroutine *DLOAD*. In the user subroutine the θ value at each integration point, which is stored in `COORDS(3)`, is expressed in degrees.

Results and discussion

The analytical solution and the Abaqus results for the CAXA8*n*, CAXA8R*n*, CAXA4*n*, and CAXA4R*n* (*n*=1, 2, 3, or 4) elements are tabulated below for a structure with $\frac{M}{I} = 1$ and dimensions $L = 6$, $R_i = 2$, and $R_o = 6$. The output locations are at points $A = (R_i, 0, 0)$, $B = (R_i, L, 0)$, $C = (R_o, 0, 0)$, and $D = (R_o, L, 0)$ on the $\theta = 0°$ plane, as shown in the figure on the previous page, and at points E, F, G, and H, which are at the corresponding locations on the $\theta = 180°$ plane. The CAXA8*n* elements match the exact solution precisely.

Variable	Exact	CAXA8*n*	CAXA8R*n*	CAXA4*n*	CAXA4R*n*
σ_{zz} at A	2	2	2.040	2.102	2.124
u_r at A	0	0	0	0	0
u_z at A	0	0	0	0	0
σ_{zz} at B	2	2	2	2.098	2.091
u_r at B	-6×10^{-7}	-6×10^{-7}	-5.927×10^{-7}	-6.000×10^{-7}	-6.015×10^{-7}
u_z at B	4×10^{-7}	4×10^{-7}	4.164×10^{-7}	3.996×10^{-7}	3.984×10^{-7}
σ_{zz} at C	6	6	5.979	5.895	5.877

Variable	Exact	CAXA8*n*	CAXA8R*n*	CAXA4*n*	CAXA4R*n*
u_r at C	-1.76×10^{-7}	-1.76×10^{-7}	-1.881×10^{-7}	-1.757×10^{-7}	-1.762×10^{-7}
u_z at C	0	0	0	0	0
σ_{zz} at D	6	6	6	5.898	5.908
u_r at D	-7.76×10^{-7}	-7.76×10^{-7}	-7.954×10^{-7}	-7.757×10^{-7}	-7.779×10^{-7}
u_z at D	1.2×10^{-6}	1.2×10^{-6}	1.211×10^{-6}	1.200×10^{-6}	1.203×10^{-6}
σ_{zz} at E	−2	−2	−2.040	−2.102	−2.124
u_r at E	0	0	0	0	0
u_z at E	0	0	0	0	0
σ_{zz} at F	−2	−2	−2	−2.098	−2.091
u_r at F	6×10^{-7}	6×10^{-7}	5.927×10^{-7}	6.000×10^{-7}	6.015×10^{-7}
u_z at F	-4×10^{-7}	-4×10^{-7}	-4.164×10^{-7}	-3.996×10^{-7}	-3.984×10^{-7}
σ_{zz} at G	−6	−6	−5.979	−5.895	−5.877
u_r at G	1.76×10^{-7}	1.76×10^{-7}	1.881×10^{-7}	1.757×10^{-7}	1.762×10^{-7}
u_z at G	0	0	0	0	0
σ_{zz} at H	−6	−6	−6	−5.898	−5.908
u_r at H	7.76×10^{-7}	7.76×10^{-7}	7.954×10^{-7}	7.757×10^{-7}	7.779×10^{-7}
u_z at H	-1.2×10^{-6}	-1.2×10^{-6}	-1.211×10^{-6}	-1.200×10^{-6}	-1.203×10^{-6}

Note:

The results are independent of *n*, the number of Fourier modes.

Figure 1 through *Figure 4* show plots of the undeformed mesh, the deformed mesh, the contours of u_r, and the contours of u_z, respectively, for the CAXA4R4 model.

Input files

ecnssfsk.inp	CAXA41 elements.
ecnssfsk.f	User subroutine *DLOAD* used in ecnssfsk.inp.
ecntsfsk.inp	CAXA42 elements.
ecntsfsk.f	User subroutine *DLOAD* used in ecntsfsk.inp.
ecnusfsk.inp	CAXA43 elements.
ecnusfsk.f	User subroutine *DLOAD* used in ecnusfsk.inp.
ecnvsfsk.inp	CAXA44 elements.
ecnvsfsk.f	User subroutine *DLOAD* used in ecnvsfsk.inp.
ecnssrsk.inp	CAXA4R1 elements.
ecnssrsk.f	User subroutine *DLOAD* used in ecnssrsk.inp.
ecntsrsk.inp	CAXA4R2 elements.
ecntsrsk.f	User subroutine *DLOAD* used in ecntsrsk.inp.

ecnusrsk.inp	CAXA4R3 elements.
ecnusrsk.f	User subroutine *DLOAD* used in ecnusrsk.inp.
ecnvsrsk.inp	CAXA4R4 elements.
ecnvsrsk.f	User subroutine *DLOAD* used in ecnvsrsk.inp.
ecnwsfsk.inp	CAXA81 elements.
ecnwsfsk.f	User subroutine *DLOAD* used in ecnwsfsk.inp.
ecnxsfsk.inp	CAXA82 elements.
ecnxsfsk.f	User subroutine *DLOAD* used in ecnxsfsk.inp.
ecnysfsk.inp	CAXA83 elements.
ecnysfsk.f	User subroutine *DLOAD* used in ecnysfsk.inp.
ecnzsfsk.inp	CAXA84 elements.
ecnzsfsk.f	User subroutine *DLOAD* used in ecnzsfsk.inp.
ecnwsrsk.inp	CAXA8R1 elements.
ecnwsrsk.f	User subroutine *DLOAD* used in ecnwsrsk.inp.
ecnxsrsk.inp	CAXA8R2 elements.
ecnxsrsk.f	User subroutine *DLOAD* used in ecnxsrsk.inp.
ecnysrsk.inp	CAXA8R3 elements.
ecnysrsk.f	User subroutine *DLOAD* used in ecnysrsk.inp.
ecnzsrsk.inp	CAXA8R4 elements.
ecnzsrsk.f	User subroutine *DLOAD* used in ecnzsrsk.inp.

Figures

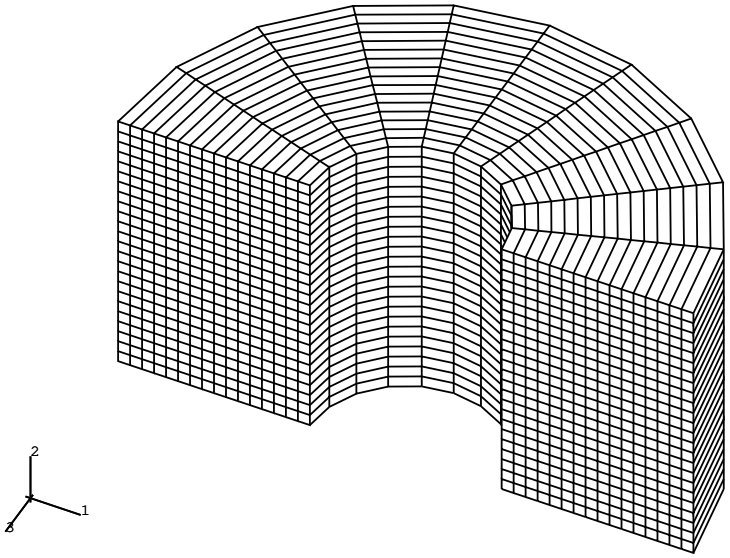


Figure 1: Undeformed mesh.

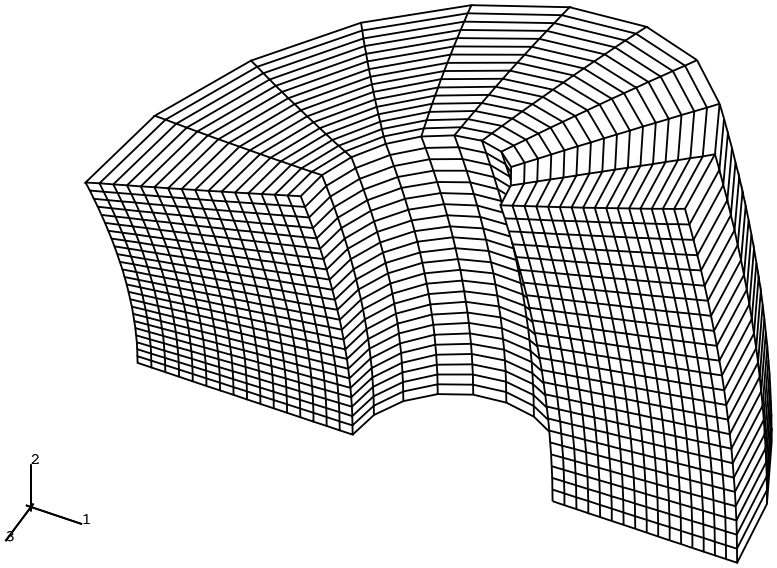


Figure 2: Deformed mesh.

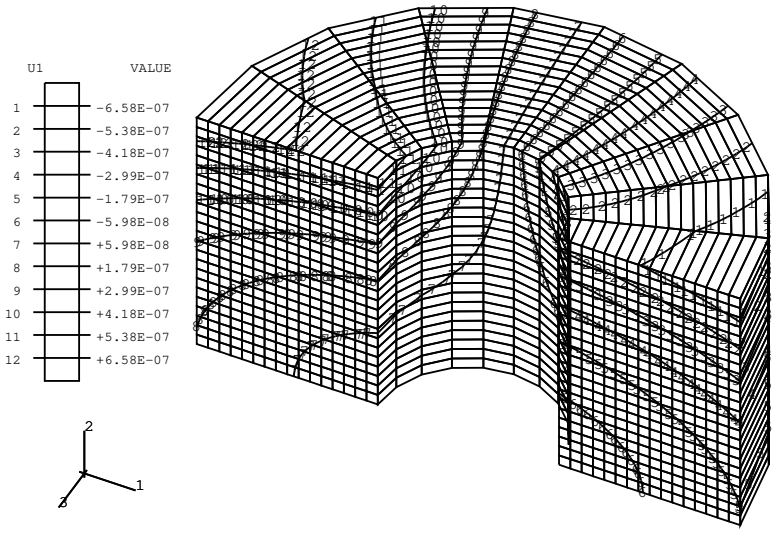


Figure 3: Contours of *r*-displacement.

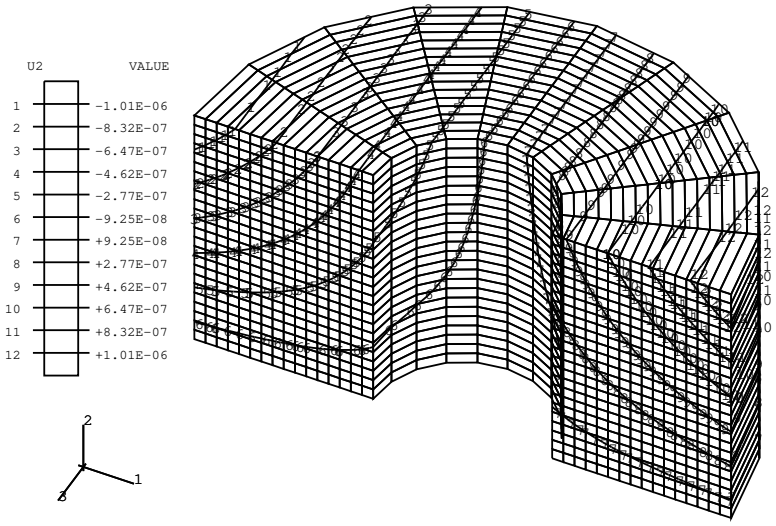


Figure 4: Contours of z-displacement.

Cylinder subjected to an asymmetric temperature field: CAXA elements

Products: Abaqus/Standard

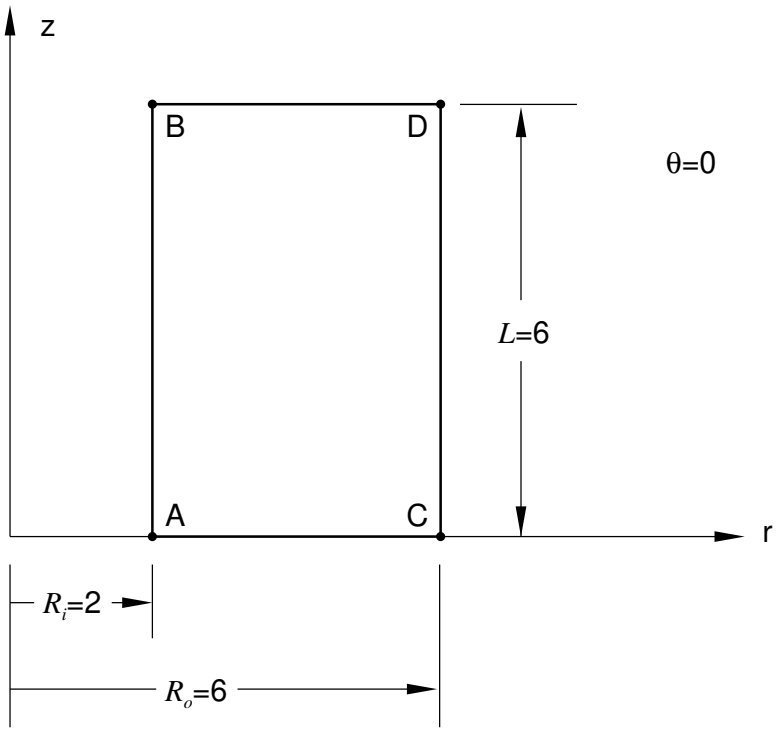
Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

CAXA4n CAXA4Rn CAXA8n CAXA8Rn (n = 1, 2, 3, 4)

Problem description



A hollow cylinder of circular cross-section, inner radius R_i, outer radius R_o, and length $2L$, is subjected to an asymmetric temperature distribution that is a linear function of the spatial coordinates:

$$T = \frac{T_o r \cos\theta}{R_o},$$

where T_o is the constant temperature at the outside surface of the cylinder at $\theta = 0°$ and r, θ, and z (see displacement solution, below) are the cylindrical coordinates. For a linear elastic material of Young's modulus E, Poisson's ratio ν, and thermal expansion coefficient α, the solution for a structure subjected to such a temperature distribution is stress-free, with displacements as follows:

$$u_r = \frac{\alpha T_o \cos\theta}{2R_o}\left(r^2 - z^2 + 2R_i R_o - R_i^2\right)$$
$$u_\theta = \frac{\alpha T_o \sin\theta}{2R_o}\left(r^2 + z^2 - 2R_i R_o - R_i^2\right)$$
$$u_z = \frac{\alpha T_o \cos\theta}{2R_o} rz.$$

Only one-half of the structure is considered, with a symmetry plane at $z = 0$. The form of the displacement solution, which is a quadratic function in both *r* and *z*, indicates that a single second-order element can model the structure adequately and yield accurate results. This problem is also solved with an 8 × 12 mesh of fully integrated first-order elements and a 16 × 24 mesh of reduced integration first-order elements.

Material: Linear elastic, Young's modulus = 30×10^6, Poisson's ratio = 0.33, coefficient of thermal expansion = 1×10^{-4}.

Boundary conditions: $u_z = 0$ on the $z = 0$ plane; $u_r = 0.06$ is applied at $r = R_i$ and $z = 0$ to eliminate the rigid body motion in the global *x*-direction. This value of u_r is obtained from the equation for u_r above.

Loading: A temperature field of the form $T = T_o r \cos\theta / R_o$ is applied by calculating the temperature at each node and defining the temperature value.

Results and discussion

The analytical solution and the Abaqus results for the CAXA8*n*, CAXA8R*n*, CAXA4*n*, and CAXA4R*n* (*n* = 1, 2, 3 or 4) elements are tabulated below for a structure with these parameters: $L = 6$, $R_i = 2$, $R_o = 6$, and $T_o = 300$. The output locations are at points $A = (R_i, 0, 0)$, $B = (R_i, L, 0)$, $C = (R_o, 0, 0)$, and $D = (R_o, L, 0)$ on the $\theta = 0°$ plane, as shown in the figure on the previous page, and at points E, F, G, and H, which are at the corresponding locations on the $\theta = 180°$ plane. While both the CAXA8*n* and CAXA8R*n* elements match the exact solution precisely with a zero state of stress, the models using the CAXA4*n* and CAXA4R*n* elements fail to predict a stress-free state, even though the displacement solutions predicted are quite reasonable. However, the CAXA4R*n* models give much more accurate results than the CAXA4*n* models. This example demonstrates that the fully integrated first-order elements do not handle bending problems very well.

Variable	Exact	CAXA8*n*	CAXA8R*n*	CAXA4*n*	CAXA4R*n*
σ_{rz} at A	0	0	0	−14071	0.0168
u_r at A	6×10^{-2}	6×10^{-2}	6×10^{-2}	6×10^{-2}	6×10^{-2}
u_z at A	0	0	0	0	0
σ_{rz} at B	0	0	0	11664	−3.2186
u_r at B	-3×10^{-2}	-3×10^{-2}	-3×10^{-2}	-2.9644×10^{-2}	-2.9999×10^{-2}
u_z at B	6×10^{-2}	6×10^{-2}	6×10^{-2}	6.0312×10^{-2}	6.0001×10^{-2}
σ_{rz} at C	0	0	0	−14076	0.0162
u_r at C	1.4×10^{-2}	1.4×10^{-2}	1.4×10^{-2}	1.3993×10^{-2}	1.4×10^{-2}
u_z at C	0	0	0	0	0
σ_{rz} at D	0	0	0	11108	−3.5190

Variable	Exact	CAXA8*n*	CAXA8R*n*	CAXA4*n*	CAXA4R*n*
u_r at D	5×10^{-2}	5×10^{-2}	5×10^{-2}	5.0306×10^{-2}	5.0001×10^{-2}
u_z at D	18×10^{-2}	18×10^{-2}	18×10^{-2}	17.95×10^{-2}	18×10^{-2}
σ_{rz} at E	0	0	0	−14071	−0.0168
u_r at E	-6×10^{-2}	-6×10^{-2}	-6×10^{-2}	-6×10^{-2}	-6×10^{-2}
u_z at E	0	0	0	0	0
σ_{rz} at F	0	0	0	−11664	3.2186
u_r at F	3×10^{-2}	3×10^{-2}	3×10^{-2}	2.9644×10^{-2}	2.9999×10^{-2}
u_z at F	-6×10^{-2}	-6×10^{-2}	-6×10^{-2}	-6.0312×10^{-2}	-6.0001×10^{-2}
σ_{rz} at G	0	0	0	14076	3.5100
u_r at G	-1.4×10^{-2}	-1.4×10^{-2}	-1.4×10^{-2}	-1.3993×10^{-2}	-1.4×10^{-2}
u_z at G	0	0	0	0	0
σ_{rz} at H	0	0	0	11108	3.5100
u_r at H	-5×10^{-2}	-5×10^{-2}	-5×10^{-2}	-5.0306×10^{-2}	-5.0001×10^{-2}
u_z at H	-18×10^{-2}	-18×10^{-2}	-18×10^{-2}	-17.95×10^{-2}	-18×10^{-2}

Note:

The results are independent of *n*, the number of Fourier modes.

Figure 1 through *Figure 4* show plots of the undeformed and deformed meshes, the applied asymmetric temperature field, the contours of u_r, and the contours of u_z, respectively, for the CAXA84 model.

Input files

ecnssfsl.inp	CAXA41 elements.
ecntsfsl.inp	CAXA42 elements.
ecnusfsl.inp	CAXA43 elements.
ecnvsfsl.inp	CAXA44 elements.
ecnssrsl.inp	CAXA4R1 elements.
ecntsrsl.inp	CAXA4R2 elements.
ecnusrsl.inp	CAXA4R3 elements.
ecnvsrsl.inp	CAXA4R4 elements.
ecnwsfsl.inp	CAXA81 elements.
ecnxsfsl.inp	CAXA82 elements.
ecnysfsl.inp	CAXA83 elements.
ecnzsfsl.inp	CAXA84 elements.
ecnwsrsl.inp	CAXA8R1 elements.
ecnxsrsl.inp	CAXA8R2 elements.
ecnysrsl.inp	CAXA8R3 elements.
ecnzsrsl.inp	CAXA8R4 elements.

Figures

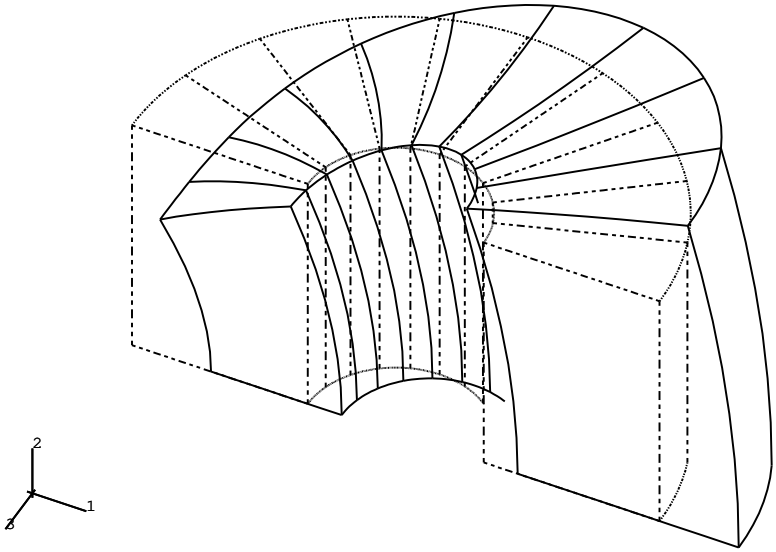


Figure 1: Deformed mesh.

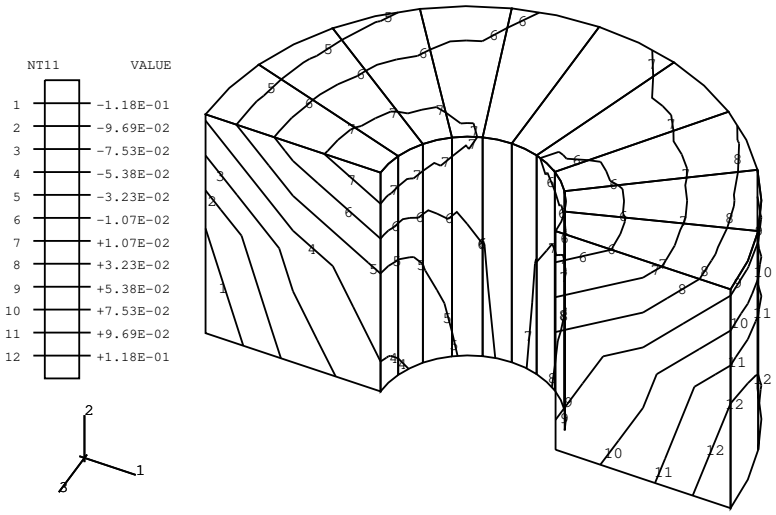


Figure 2: Applied temperature field.

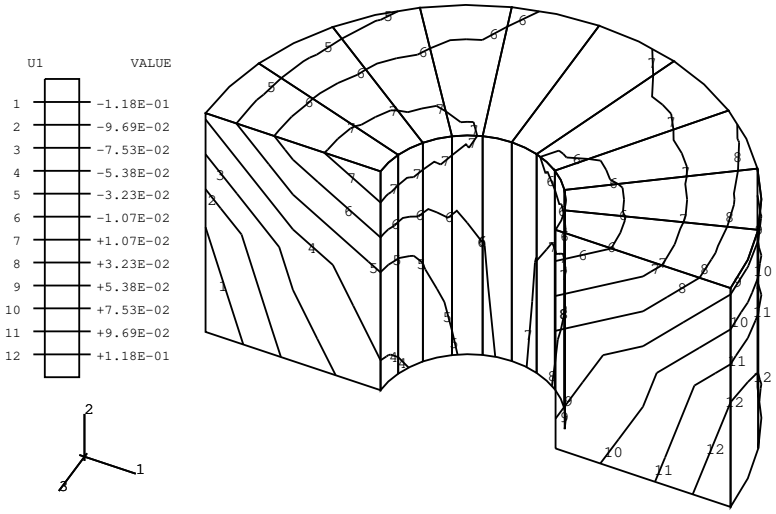


Figure 3: Contours of r-displacement.

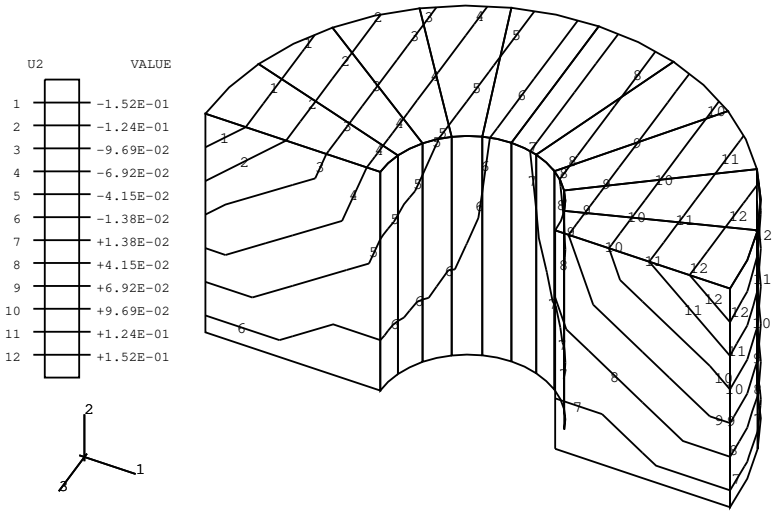


Figure 4: Contours of z-displacement.

Cylinder subjected to asymmetric pressure loads: CAXA elements

Products: Abaqus/Standard

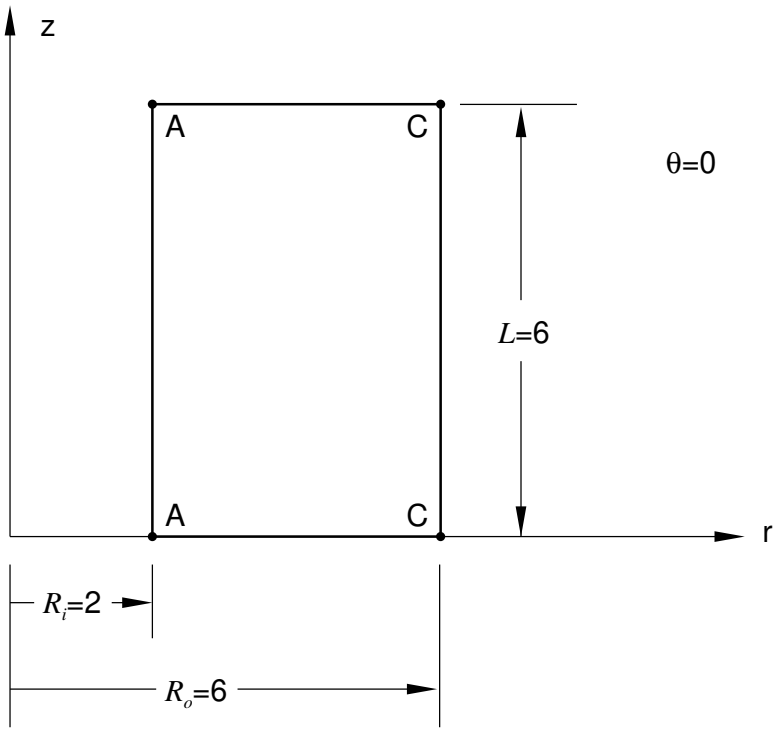
Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

CAXA4n CAXA4Rn CAXA8n CAXA8Rn (n = 1, 2, 3, 4)

Problem description



A hollow cylinder of circular cross-section, inner radius R_i, outer radius R_o, and length $2L$ is subjected to both internal and external pressure loads that are asymmetric. The pressure stresses take the following forms:

$\sigma_{rr} = -p\cos\theta$ at $r = R_o$ and $\sigma_{rr} = -\frac{R_o}{R_i}p\cos\theta$ at $r = R_i$, where p is a pressure value and r and θ are the cylindrical coordinates. Assuming plane strain conditions and a linear elastic material with Young's modulus E and Poisson's ratio ν, the small-displacement solutions for stress and displacement are as follows:

$$
\begin{aligned}
\sigma_{rr} &= A\left[r + \frac{R_i^2 R_o^2}{r^3} + \frac{(3-2\nu)}{(1-2\nu)}\frac{\left(R_o^2+R_i^2\right)}{r}\right]\cos\theta \\
\sigma_{\theta\theta} &= A\left[3r - \frac{R_i^2 R_o^2}{r^3} - \frac{\left(R_o^2+R_i^2\right)}{r}\right]\cos\theta \\
\sigma_{zz} &= \nu\left(\sigma_{rr}+\sigma_{\theta\theta}\right) \\
\sigma_{r\theta} &= A\left[r + \frac{R_i^2 R_o^2}{r^3} - \frac{\left(R_o^2+R_i^2\right)}{r}\right]\sin\theta \\
u_r &= \frac{1+\nu}{2E}A\left[(1-4\nu)r^2 - \frac{R_i^2 R_o^2}{r^2} + 2\frac{(3-4\nu)}{(1-2\nu)}\left(R_o^2+R_i^2\right)\ln r\right]\cos\theta \\
u_\theta &= \frac{1+\nu}{2E}A\left[(5-4\nu)r^2 - \frac{R_i^2 R_o^2}{r^2} - 2\frac{(3-4\nu)}{(1-2\nu)}\left(R_o^2+R_i^2\right)\ln r - 2\frac{\left(R_o^2+R_i^2\right)}{(1-2\nu)}\right]\sin\theta,
\end{aligned}
$$

where

$$
A = -(1-2\nu)\frac{R_o p}{4(1-\nu)\left(R_o^2+R_i^2\right)}.
$$

Only a slice of the cylinder is considered. Plane strain conditions are applied by setting $u_z = 0$ everywhere. In the *r*-direction 10 elements are used in the second-order element models. In models using the first-order elements, 20 and 40 elements are used in the full- and reduced-integration models, respectively.

Material: Linear elastic, Young's modulus = 30×10^6, Poisson's ratio = 0.3.

Boundary conditions: $u_z = 0$ everywhere; $u_r = -9.9854 \times 10^{-4}$ at $r = R_i$ and $\theta = 0°$, as obtained from the equation for u_r above. These constraints eliminate the rigid body motions in the global *z*- and *x*-directions, respectively.

Loading: The asymmetric pressure loads are prescribed by applying the appropriate nonuniform distributed load types on the inside and outside surfaces of the cylinder and defining the pressure stress equations for σ_{rr} in user subroutine *DLOAD*. In the user subroutine, the θ value at each integration point, which is stored in `COORDS(3)`, is expressed in degrees.

Results and discussion

The analytical solution and the Abaqus results for the CAXA8*n*, CAXA8R*n*, CAXA4*n*, and CAXA4R*n* (*n* = 1, 2, 3 or 4) elements are tabulated below for a cylinder with these parameters: $L = 6$, $R_i = 2$, $R_o = 6$, and $P = 10 \times 10^3$. The output locations are at points $A = (R_i, z, 0)$ and $C = (R_o, z, 0)$ on the $\theta = 0°$ plane, where *z* can be any value along lines AA and CC in the figure shown on the previous page since the solution is independent of *z*, and at points *E* and *G*, which are at the corresponding locations on the $\theta = 180°$ plane. The solutions predicted by Abaqus agree well with the exact solution. Closer agreement is anticipated if a denser mesh is used.

Variable	Exact	CAXA8*n*	CAXA8R*n*	CAXA4*n*	CAXA4R*n*
σ_{rr} at A	−30000.0	−29610.0	−29760.0	−28617.0	−29132.0
σ_{zz} at A	−7890.4	−7702.7	−7849.6	−7885.1	−7722.9
$\sigma_{\theta\theta}$ at A	6089.6	6268.2	5973.4	6034.6	5729.2

Variable	Exact	CAXA8*n*	CAXA8R*n*	CAXA4*n*	CAXA4R*n*
$\sigma_{r\theta}$ at A	0.0	0.0	0.0	0.0	0.0
u_r at A	-9.9854×10^{-4}	-9.9854×10^{-4}	-9.9854×10^{-4}	-9.9854×10^{-4}	-9.9854×10^{-4}
σ_{rr} at C	−10000.0	−9988.9	−9992.4	−10101.0	−10205.0
σ_{zz} at C	−3969.9	−3964.4	−3967.9	−3952.2	−3978.2
$\sigma_{\theta\theta}$ at C	−2029.9	−2024.3	−2031.5	−2013.4	−1902.1
$\sigma_{r\theta}$ at C	0.0	0.0	0.0	0.0	0.0
u_r at C	-2.9222×10^{-3}	-2.9222×10^{-3}	-2.9222×10^{-3}	-2.9207×10^{-3}	-2.9221×10^{-3}
σ_{rr} at E	30000.0	29610.0	29760.0	28617.0	29132.0
σ_{zz} at E	7890.4	7702.7	7849.6	7885.1	7722.9
$\sigma_{\theta\theta}$ at E	−6089.6	−6268.2	−5973.4	−6034.6	−5729.2
$\sigma_{r\theta}$ at E	0.0	0.0	0.0	0.0	0.0
u_r at E	9.9854×10^{-4}	9.9854×10^{-4}	9.9854×10^{-4}	9.9854×10^{-4}	9.9854×10^{-4}
σ_{rr} at G	10000.0	9988.9	9992.4	10101.0	10067.0
σ_{zz} at G	3969.9	3964.4	3967.9	3952.2	3978.2
$\sigma_{\theta\theta}$ at G	2029.9	2024.3	2031.5	2013.4	1987.9
$\sigma_{r\theta}$ at G	0.0	0.0	0.0	0.0	0.0
u_r at G	2.9222×10^{-3}	2.9222×10^{-3}	2.9222×10^{-3}	2.9207×10^{-3}	2.9221×10^{-3}

Note:

The results are independent of *n*, the number of Fourier modes. The u_θ variable is not compared, since u_θ is treated as an internal variable in these elements and is not available for output. The accuracy of u_θ may be assumed to be comparable to the accuracy of u_r.

Figure 1 through *Figure 4* show plots of the undeformed mesh, the deformed mesh, the contours of σ_{rr}, and the contours of u_r, respectively, for the CAXA8R3 model.

Input files

ecnssfsm.inp	CAXA41 elements.
ecnssfsm.f	User subroutine *DLOAD* used in ecnssfsm.inp.
ecntsfsm.inp	CAXA42 elements.
ecntsfsm.f	User subroutine *DLOAD* used in ecntsfsm.inp.
ecnusfsm.inp	CAXA43 elements.
ecnusfsm.f	User subroutine *DLOAD* used in ecnusfsm.inp.
ecnvsfsm.inp	CAXA44 elements.
ecnvsfsm.f	User subroutine *DLOAD* used in ecnvsfsm.inp.
ecnssrsm.inp	CAXA4R1 elements.
ecnssrsm.f	User subroutine *DLOAD* used in ecnssrsm.inp.

ecntsrsm.inp	CAXA4R2 elements.
ecntsrsm.f	User subroutine *DLOAD* used in ecntsrsm.inp.
ecnusrsm.inp	CAXA4R3 elements.
ecnusrsm.f	User subroutine *DLOAD* used in ecnusrsm.inp.
ecnvsrsm.inp	CAXA4R4 elements.
ecnvsrsm.f	User subroutine *DLOAD* used in ecnvsrsm.inp.
ecnwsfsm.inp	CAXA81 elements.
ecnwsfsm.f	User subroutine *DLOAD* used in ecnwsfsm.inp.
ecnxsfsm.inp	CAXA82 elements.
ecnxsfsm.f	User subroutine *DLOAD* used in ecnxsfsm.inp.
ecnysfsm.inp	CAXA83 elements.
ecnysfsm.f	User subroutine *DLOAD* used in ecnysfsm.inp.
ecnzsfsm.inp	CAXA84 elements.
ecnzsfsm.f	User subroutine *DLOAD* used in ecnzsfsm.inp.
ecnwsrsm.inp	CAXA8R1 elements.
ecnwsrsm.f	User subroutine *DLOAD* used in ecnwsrsm.inp.
ecnxsrsm.inp	CAXA8R2 elements.
ecnxsrsm.f	User subroutine *DLOAD* used in ecnxsrsm.inp.
ecnysrsm.inp	CAXA8R3 elements.
ecnysrsm.f	User subroutine *DLOAD* used in ecnysrsm.inp.
ecnzsrsm.inp	CAXA8R4 elements.
ecnzsrsm.f	User subroutine *DLOAD* used in ecnzsrsm.inp.

Figures

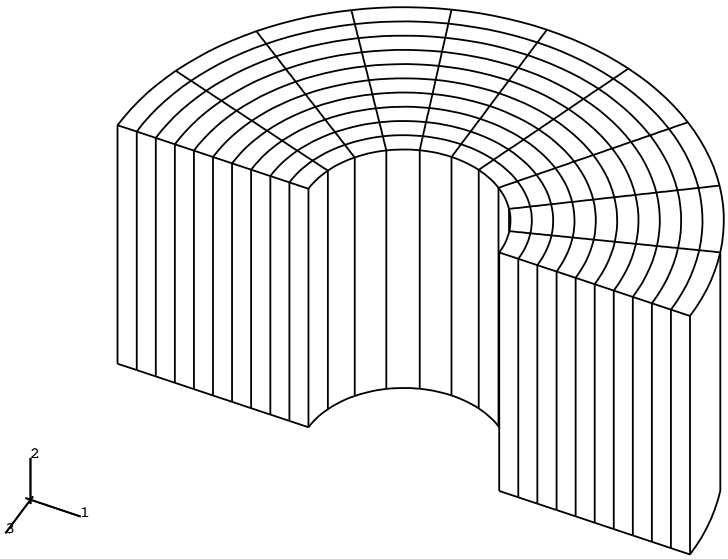


Figure 1: Undeformed mesh.

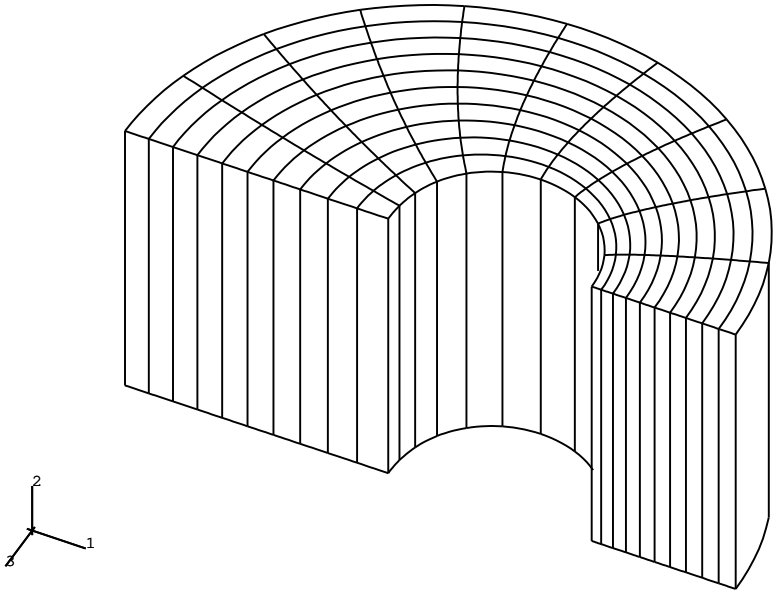


Figure 2: Deformed mesh.

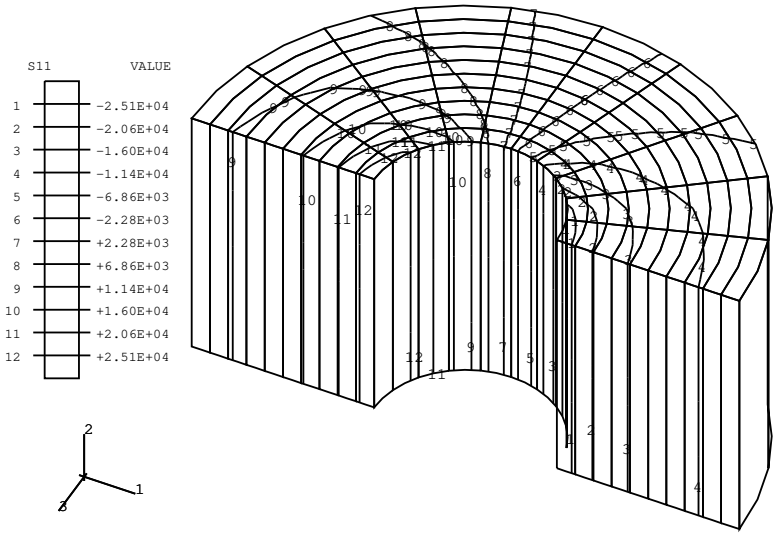


Figure 3: Contours of radial stress.

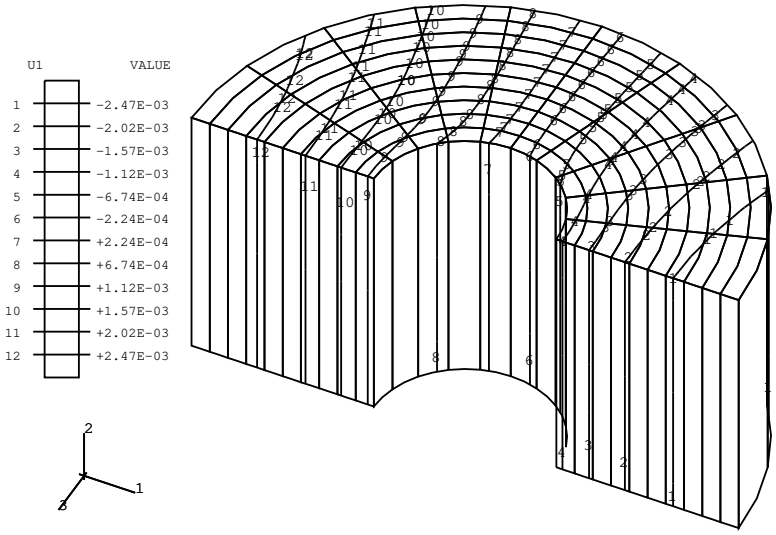


Figure 4: Contours of *r*-displacement.

Cylinder subjected to an asymmetric pore pressure field: CAXA elements

Products: Abaqus/Standard

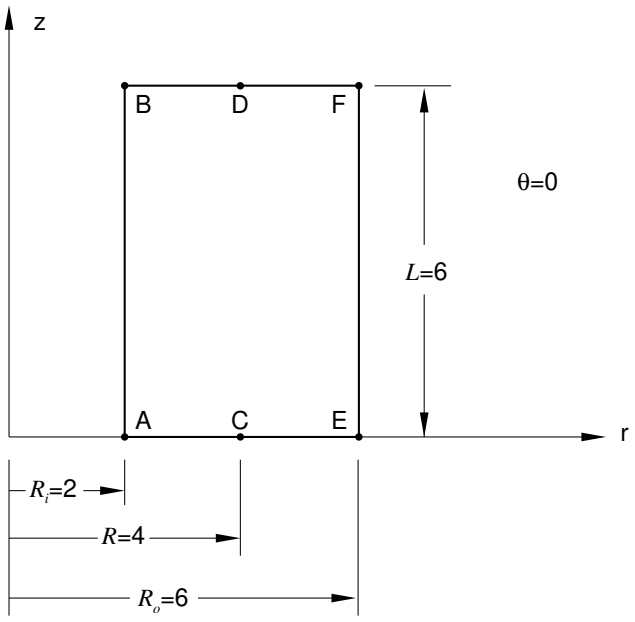
Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

CAXA8Pn CAXA8RPn (n = 1, 2, 3, 4)

Problem description



A hollow cylindrical soil column of circular cross-section, inner radius R_i, outer radius R_o, and length $2L$ is subjected to an asymmetric pore pressure distribution of the form

$$U = \frac{U_o r \cos\theta}{R_o},$$

where U_o is the constant pore pressure at the outside surface of the cylinder at $\theta = 0°$ and r and θ are the cylindrical coordinates. The presence of pore pressure gradients in the radial and circumferential directions causes the pore fluid in the soil to flow in these directions, and bending of the cylinder results. This is a coupled problem in which the stress equilibrium and fluid continuity equations must be solved simultaneously with the pore pressure CAXA elements. For illustration purposes we consider only the steady-state coupled problem, and we assume that the material is linear with constant permeability and is made up of incompressible grains and fluid. The results predicted by the pore pressure CAXA models will be compared with those predicted by the corresponding three-dimensional model.

Only one-half of the structure is considered, with a symmetry plane at $z = 0$. A mesh convergence study indicates that a single second-order CAXA element can model the structure adequately and yield accurate results. However, two elements are used in the radial direction so that direct comparison of results obtained with the three-dimensional model can be made. In the three-dimensional model the C3D20P element is used in a finite element mesh with 2 elements in the radial direction, 1 in the axial direction, and 12 in the circumferential direction. To facilitate comparison of results with the CAXA models, all nodes in the three-dimensional model are transformed to a local cylindrical system, and a cylindrical orientation is applied to the material so that displacement, stress, and strain components are output in the same cylindrical system.

Material: Linear elastic, Young's modulus = 1×10^8, Poisson's ratio = 0.3, permeability = 1×10^{-5}, initial void ratio = 1.0 everywhere.

Boundary conditions: $u_z = 0$ on the $z = 0$ plane; $u_r = 0$ is applied at $r = R_i$ and $z = 0$ to eliminate the rigid body motion in the global x-direction.

Loading: A pore pressure field of the form $U = U_o r \cos\theta / R_o$ is applied. The pore pressure at each corner node on the inside and outside walls of the cylinder is calculated, and the pore pressure values are prescribed as boundary conditions via degree of freedom 8.

Results and discussion

The results obtained with the CAXA8P*n* and CAXA8RP*n* (n = 1, 2, 3 or 4) elements and those obtained with the C3D20P elements are tabulated below for a structure with these parameters: $L = 6$, $R_i = 2$, $R_o = 6$, and $U_o = 3 \times 10^6$. The output locations are at points $A = (R_i, 0, 0)$, $B = (R_i, L, 0)$, $C = (4, 0, 0)$, $D = (4, L, 0)$, $E = (R_o, 0, 0)$, and $F = (R_o, L, 0)$ on the $\theta = 0°$ plane, as shown in the figure on the previous page. Results that are exactly equal and opposite to those shown below are obtained at the same locations on the $\theta = 180°$ plane. It is apparent that the results of the CAXA models match closely with the results of the three-dimensional model. The stress solution, which is shown in the table below, reveals that the effective stress components are identical to the pore pressure everywhere so that the total stress is zero everywhere in the cylinder. The results obtained from the CAXA models are independent of n, the number of Fourier modes, and appear to be more accurate than the three-dimensional model because the applied asymmetric pore pressure field can be prescribed precisely in the CAXA models. In the three-dimensional model more elements are needed in the θ-direction to get results with higher accuracy. Note the similarity between the solution to this problem and the asymmetric temperature analysis described in *Cylinder subjected to an asymmetric temperature field: CAXA elements*.

Variable	C3D20P	CAXA8P*n*	CAXA8RP*n*
u_r at A	0	0	0
u_z at A	0	0	0
U at A	1×10^6	1×10^6	1×10^6
σ_{rr} at A	9.9066×10^5	1×10^6	1×10^6
σ_{zz} at A	9.9397×10^5	1×10^6	1×10^6
$\sigma_{\theta\theta}$ at A	9.9751×10^5	1×10^6	1×10^6
u_r at B	-3.5791×10^{-2}	-3.6×10^{-2}	-3.6×10^{-2}
u_z at B	2.3853×10^{-2}	2.4×10^{-2}	2.4×10^{-2}
U at B	1×10^6	1×10^6	1×10^6

Variable	C3D20P	CAXA8P*n*	CAXA8RP*n*
σ_{rr} at B	9.9066×10^{5}	1×10^{6}	1×10^{6}
σ_{zz} at B	9.9397×10^{5}	1×10^{6}	1×10^{6}
$\sigma_{\theta\theta}$ at B	9.9751×10^{5}	1×10^{6}	1×10^{6}
u_r at C	1.1926×10^{-2}	1.2×10^{-2}	1.2×10^{-2}
u_z at C	0	0	0
U at C	1.9987×10^{6}	2×10^{6}	2×10^{6}
σ_{rr} at C	1.9944×10^{6}	2×10^{6}	2×10^{6}
σ_{zz} at C	1.9947×10^{6}	2×10^{6}	2×10^{6}
$\sigma_{\theta\theta}$ at C	2.0038×10^{6}	2×10^{6}	2×10^{6}
u_r at D	-2.3864×10^{-2}	-2.4×10^{-2}	-2.4×10^{-2}
u_z at D	4.7711×10^{-2}	4.8×10^{-2}	4.8×10^{-2}
U at D	1.9987×10^{6}	2×10^{6}	2×10^{6}
σ_{rr} at D	1.9943×10^{6}	2×10^{6}	2×10^{6}
σ_{zz} at D	1.9945×10^{6}	2×10^{6}	2×10^{6}
$\sigma_{\theta\theta}$ at D	2.0036×10^{6}	2×10^{6}	2×10^{6}
u_r at E	3.1819×10^{-2}	3.2×10^{-2}	3.2×10^{-2}
u_z at E	0	0	0
U at E	3×10^{6}	3×10^{6}	3×10^{6}
σ_{rr} at E	3.004×10^{6}	3×10^{6}	3×10^{6}
σ_{zz} at E	2.9961×10^{6}	3×10^{6}	3×10^{6}
$\sigma_{\theta\theta}$ at E	3.0105×10^{6}	3×10^{6}	3×10^{6}
u_r at F	-3.9718×10^{-2}	-0.4×10^{-2}	-0.4×10^{-2}
u_z at F	7.1581×10^{-2}	7.2×10^{-2}	7.2×10^{-2}
U at F	3×10^{6}	3×10^{6}	3×10^{6}
σ_{rr} at F	3.004×10^{6}	3×10^{6}	3×10^{6}
σ_{zz} at F	2.9965×10^{6}	3×10^{6}	3×10^{6}
$\sigma_{\theta\theta}$ at F	3.0105×10^{6}	3×10^{6}	3×10^{6}

Figure 1 through *Figure 4* show plots of the undeformed and deformed meshes, the applied asymmetric pore pressure field, the contours of u_r, and the contours of u_z, respectively, in the CAXA8P4 model.

Input files

ecnwpfsn.inp	CAXA8P1 elements.
ecnxpfsn.inp	CAXA8P2 elements.
ecnypfsn.inp	CAXA8P3 elements.

ecnzpfsn.inp	CAXA8P4 elements.
ecnwprsn.inp	CAXA8RP1 elements.
ecnxprsn.inp	CAXA8RP2 elements.
ecnyprsn.inp	CAXA8RP3 elements.
ecnzprsn.inp	CAXA8RP4 elements.
eref3ksn.inp	C3D20P elements (reference solution).

Figures

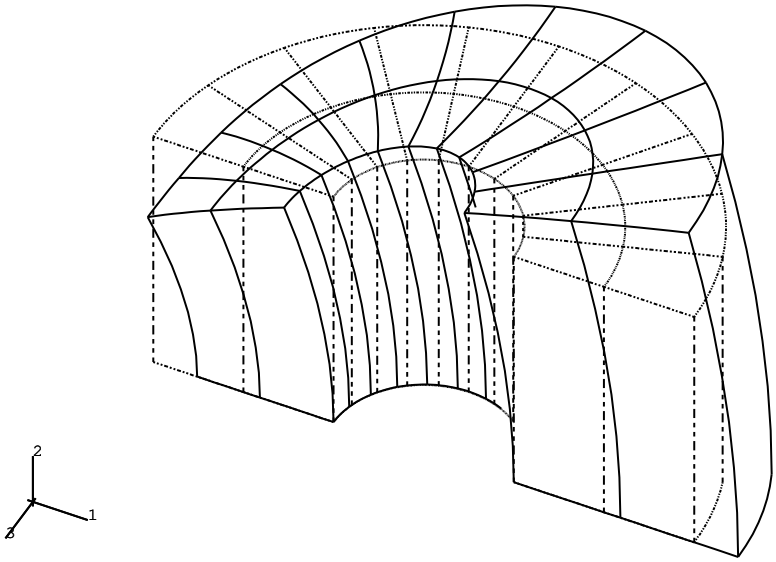


Figure 1: Deformed mesh.

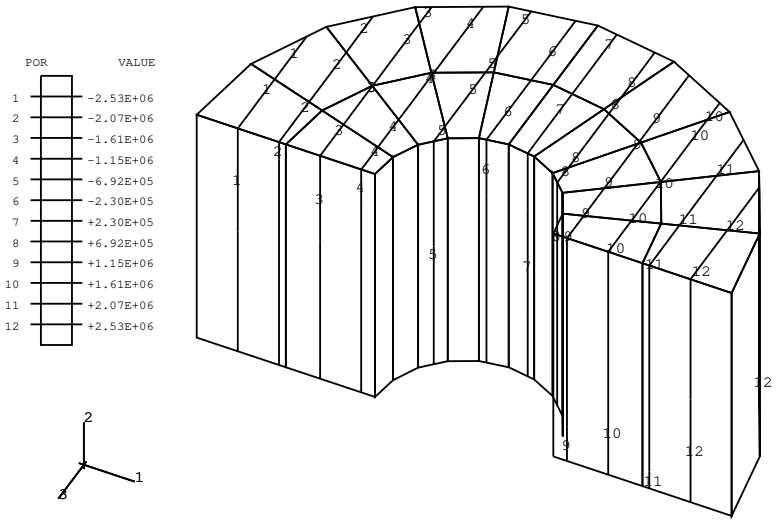


Figure 2: Contours of pore pressure.

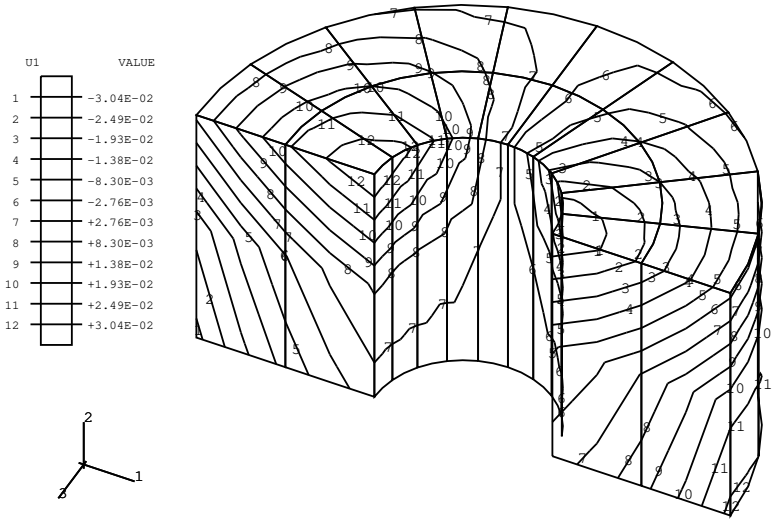


Figure 3: Contours of r-displacement.

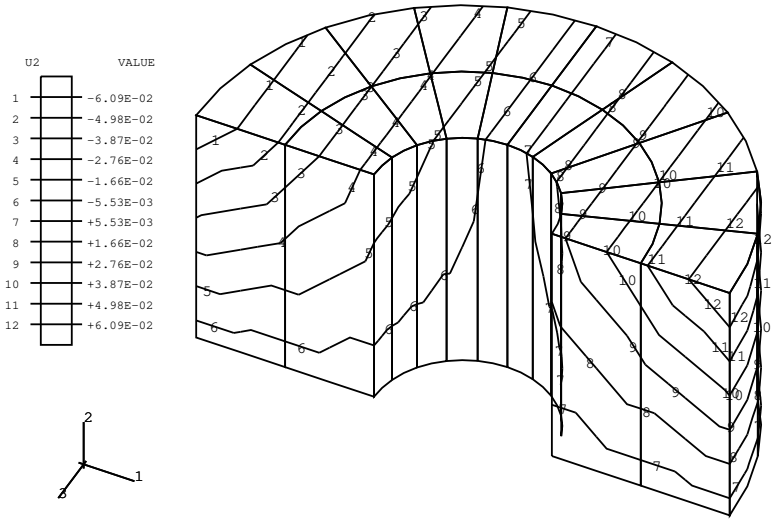


Figure 4: Contours of z-displacement.

Modal dynamic and transient dynamic analysis with CAXA and SAXA elements

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Modal dynamic and transient dynamic analysis

Elements tested

CAXA4n CAXA4Rn CAXA4RHn CAXA8n CAXA8Rn CAXA8RHn SAXA1n SAXA2n (n = 2, 3, 4)

Problem description

A cantilever pipe 100 units long with an outer radius of 1.2675 units and a wall thickness of 0.2 units subjected to tip loading is analyzed. The pipe is modeled with all the elements listed above. The first-order, fully integrated CAXA model consists of 2×20 elements in the mesh, while the CAXA4R*n* and the CAXA4RH*n* models consist of 4×40 elements in the mesh. The second-order CAXA models use 20 elements along the length of the pipe. The first-order SAXA*n* model uses 20 elements along the length of the pipe, while 10 elements are used in the SAXA2*n* model. The material behavior is assumed to be isotropic elastic with a Young's modulus of 30.E6 and Poisson's ratio of 0.3.

The modal dynamic and steady-state dynamic procedures, the direct-solution steady-state procedure, the subspace-based steady-state procedure, and the transient dynamic procedure are used in the verification tests. A sinusoidal load with a maximum amplitude of 1.0E4 units is applied to the tip of the cantilever pipe. The concentrated load is split in two, with half applied to the midside nodes in each of the $\theta = 0°$ and $\theta = 180°$ planes on the loaded end of the pipe. All the nodes on one end of the pipe are fixed. To avoid any deformation through the wall thickness in the CAXA model caused by the application of concentrated loads on the loaded end, the radial displacements at the midside nodes are constrained to be equal to the average radial motion of the nodes at the inside and outside radii.

Results and discussion

The results of the tests agree well with the results obtained by modeling the cantilever pipe with beam elements having pipe cross-sections.

Input files

ecntsfdyn.inp	CAXA42 elements.
ecnusfdyn.inp	CAXA43 elements.
ecnvsfdyn.inp	CAXA44 elements.
ecntsrdyn.inp	CAXA4R2 elements.
ecnusrdyn.inp	CAXA4R3 elements.
ecnvsrdyn.inp	CAXA4R4 elements.
ecnxsfdyn.inp	CAXA82 elements.
ecnysfdyn.inp	CAXA83 elements.
ecnzsfdyn.inp	CAXA84 elements.

ecnxsrdyn.inp	CAXA8R2 elements.
ecnysrdyn.inp	CAXA8R3 elements.
ecnzsrdyn.inp	CAXA8R4 elements.
esntsxdyn.inp	SAXA12 elements.
esnusxdyn.inp	SAXA13 elements.
esnvsxdyn.inp	SAXA14 elements.
esnxsxdyn.inp	SAXA22 elements.
esnysxdyn.inp	SAXA23 elements.
esnzsxdyn.inp	SAXA24 elements.

Random response analysis

Elements tested

CAXA4n CAXA8n SAXA1n SAXA2n (n = 2, 3, 4)

Problem description

The cantilever pipe described in the previous section is used in these verification tests. A white noise power spectral density is used to describe the applied ground accelerations. The material definition is assumed to be isotropic elastic. The values are not important.

Since random response analysis is a modal-based procedure, an eigenfrequency extraction step is required to obtain the mode shapes and natural frequencies of the system. The first ten modes are used in the random response steps with a damping ratio of 0.01 for each mode. The base motion is applied only to degree of freedom 1.

Results and discussion

The results of the analysis compare well with the results obtained by modeling the cantilever pipe with beam elements having pipe cross-sections.

Input files

ecntsfrr.inp	CAXA42 elements.
ecnusfrr.inp	CAXA43 elements.
ecnvsfrr.inp	CAXA44 elements.
ecnxsfrr.inp	CAXA82 elements.
ecnysfrr.inp	CAXA83 elements.
ecnzsfrr.inp	CAXA84 elements.
esntsxrr.inp	SAXA12 elements.
esnusxrr.inp	SAXA13 elements.
esnvsxrr.inp	SAXA14 elements.
esnxsxrr.inp	SAXA22 elements.
esnysxrr.inp	SAXA23 elements.
esnzsxrr.inp	SAXA24 elements.

Response spectrum analysis

Elements tested

CAXA42 CAXA82

Problem description

The model consists of a cylinder 300 units in length with an outer radius of 2 units. The finite element mesh consists of a single element that has nodes lying on the axis from each of the planes forming the element. The nodes on the axis are tied such that the element can simulate a solid cylinder. The material properties are assumed to be isotropic elastic. The values are not important.

The spectrum of peak displacement values as a function of frequency and damping ratio is specified in the spectrum step, and the base motion is applied in directions 1 (r-direction) and 2 (z-direction) using the response spectrum.

Results and discussion

The results of the analysis compare well with the results obtained by modeling the cylinder with beam elements.

Input files

ecntsfrs.inp	CAXA42 elements.
ecnxsfrs.inp	CAXA82 elements.

Modal dynamic analysis with baseline correction

Elements tested

CAXA4H2

Problem description

This problem is similar to the verification problem *pmodbas3.inp* using CAX4H elements described in *Modal dynamic analysis with baseline correction*. CAXA4H2 elements are used in the present verification test. The test illustrates the use of baseline correction and base motion for CAXA elements.

The structure analyzed is a cylinder made of rubberlike material. An 8 × 8 mesh of CAXA4H2 elements is employed. The nodes on the axis of the cylinder are constrained such that they do not move away from the axis after deformation.

The structure is preloaded statically in compression in the axial direction by a rigid platen. The response to applied axial excitation at the rigid surface is sought. The acceleration records are the same as those used in the problem *pmodbase.inp* (see *Modal dynamic analysis with baseline correction*).

Results and discussion

The results agree with those obtained with the verification problem *pmodbas3.inp*.

Input files

ecntsfbc.inp	CAXA4H2 elements.

Free cylinder: axisymmetric vibration

Elements tested

CAXA4n CAXA8n (n = 1, 2, 3, 4)

Problem description

This problem is similar to the problem described in *FV41: Free cylinder: axisymmetric vibration*, where axisymmetric elements are used.

The axisymmetric behavior is simulated by imposing the condition that the radial and axial displacements of the nodes on planes other than the $\theta = 0°$ plane be the same as the nodes on the $\theta = 0°$ plane.

Results and discussion

The results obtained by using CAXA elements compare well with those described in *FV41: Free cylinder: axisymmetric vibration*.

Input files

ecnssffv41.inp	CAXA41 elements.
ecntsffv41.inp	CAXA42 elements.
ecnusffv41.inp	CAXA43 elements.
ecnvsffv41.inp	CAXA44 elements.
ecnwsffv41.inp	CAXA81 elements.
ecnxsffv41.inp	CAXA82 elements.
ecnysffv41.inp	CAXA83 elements.
ecnzsffv41.inp	CAXA84 elements.

Thick hollow sphere: uniform radial vibration

Elements tested

CAXA4n CAXA8n (n = 1, 2, 3, 4)

Problem description

This problem is similar to the problem described in *FV42: Thick hollow sphere: uniform radial vibration*, where axisymmetric elements are used.

Results and discussion

The results obtained by using CAXA elements compare well with those described in *FV42: Thick hollow sphere: uniform radial vibration*.

Input files

ecnssffv42.inp	CAXA41 elements.
ecntsffv42.inp	CAXA42 elements.
ecnusffv42.inp	CAXA43 elements.
ecnvsffv42.inp	CAXA44 elements.
ecnwsffv42.inp	CAXA81 elements.

ecnxsffv42.inp	CAXA82 elements.
ecnysffv42.inp	CAXA83 elements.
ecnzsffv42.inp	CAXA84 elements.

Simple load tests for thermal-electrical elements

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

DC1D2E DC1D3E
DC2D3E DC2D4E DC2D6E DC2D8E
DCAX3E DCAX4E DCAX6E DCAX8E
DC3D4E DC3D6E DC3D8E DC3D10E DC3D15E DC3D20E

Problem description

The problem illustrated in *Figure 1* consists of a 1-m-long conductor through which a constant current density of 6.58E5 Am^{-2} is established by creating a potential difference across the ends of the conductor or by prescribing a concentrated current. The electrical energy generated by the flow of current is converted into heat, which results in a temperature distribution through the conductor. Only a steady-state solution is considered for each test. A reasonable mesh is used in each case to obtain the quadratic distribution of heat.

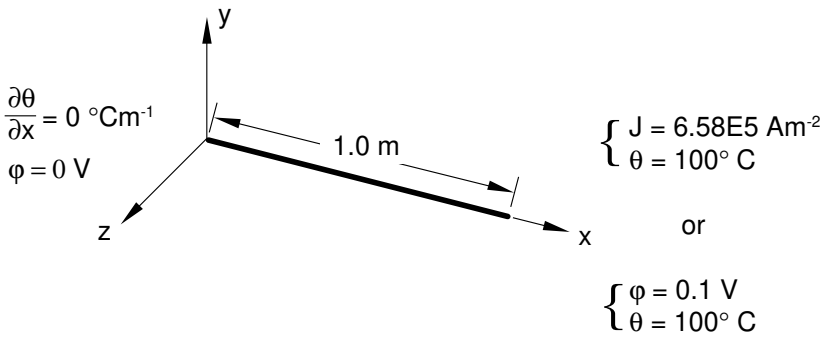


Figure 1: Model of conductor.

Material: Thermal conductivity = 45 W/m°C; electrical conductivity = 6.58E6 1/Ω m.

Boundary conditions: Zero potential ($\varphi = 0$ V) and zero temperature gradient ($\partial\theta/\partial x = 0°Cm^{-1}$) at $x = 0$ m.

Potential $\varphi = 0.1$ V and temperature $\theta = 100°C$, or current density of 6.58E5 Am^{-2} and temperature $\theta = 100°C$ at $x = 1$ m.

With these boundary conditions the problem is one-dimensional. It is assumed that all electrical energy is converted into heat.

Reference solution

In this uniaxial problem the exact solution for the temperature is of the form $\theta = C_0 \frac{x^2}{2} + C_1 x + C_2$, where C_0, C_1, and C_2 are real constants. Application of the above material properties and boundary conditions leads to the exact solution

$$\theta = C_0 \frac{x^2}{2} + 100 - \frac{C_0}{2},$$

where $C_0 = -1462.2$.

Results and discussion

The tests are composed of three steps.

In Step 1 the proper temperature boundary conditions are applied, and the flow of current is obtained by a potential difference between the two ends of the conductor. The coupled thermal-electrical procedure is used to obtain the desired temperature distribution across the conductor. For first-order elements the results are a function of *y* and *z* when the mesh generated is skewed in the *x*–*y* plane and/or the *x*–*z* plane. For the different test cases studied, the temperature may vary by as much as 3% in the *y*–*z* plane for a given *x*-value. Therefore, special care is needed when using triangular and tetrahedral elements. The exact solution is recovered in most test cases, with a maximum deviation of 1.5% from the exact solution observed with the DC3D6E elements. For second-order elements the exact results are obtained since the results are at most a quadratic function of the variable *x*. Moreover, skewed meshes do not affect the results.

Step 2 is a heat transfer step in which the conductor is allowed to cool down.

Step 3 invokes a coupled thermal-electrical procedure in which the same amount of electrical energy as that of Step 1 is provided to the specimen. However, energy is now supplied by specifying a prescribed current at $x = 1$ m instead of a potential of 0.1 V. Here again, the temperature results are identical to those obtained in Step 1, and the potential distribution that served as input for Step 1 is retrieved as output in this step.

Input files

eca3vfsj.inp	DCAX3E elements.
eca4vfsj.inp	DCAX4E elements.
eca6vfsj.inp	DCAX6E elements.
eca8vfsj.inp	DCAX8E elements.
ec12vfsj.inp	DC1D2E elements.
ec13vfsj.inp	DC1D3E elements.
ec23vfsj.inp	DC2D3E elements.
ec24vfsj.inp	DC2D4E elements.
ec26vfsj.inp	DC2D6E elements.
ec28vfsj.inp	DC2D8E elements.
ec34vfsj.inp	DC3D4E elements.
ec36vfsj.inp	DC3D6E elements.
ec38vfsj.inp	DC3D8E elements.
ec3avfsj.inp	DC3D10E elements.
ec3fvfsj.inp	DC3D15E elements.
ec3kvfsj.inp	DC3D20E elements.

Hydrostatic fluid elements

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Features tested

This section provides basic verification tests for the fluid elements that are generated in Abaqus/Standard when the fluid cavity capability is used.

Elements tested

F2D2 F3D3 F3D4 FAX2

Problem description

For the two-dimensional and three-dimensional cases, a “block” of incompressible fluid is subjected to a system of loads, as shown in *Figure 1*. The downward force causes the fluid to compress vertically and expand horizontally, while maintaining the original fluid volume (since the fluid is incompressible). The spring resists the horizontal expansion of the fluid, thus generating internal pressure in the fluid. The first axisymmetric problem is similar: the fluid volume is now a cylinder, compressed axially, with a spring resisting the radial expansion. In the second axisymmetric problem the pressure inside the fluid is specified. No external loading is applied, and the “walls” bounding the fluid are fixed.

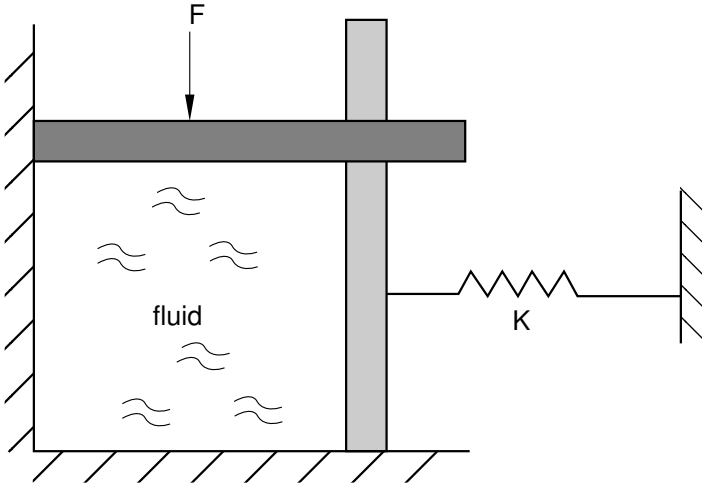


Figure 1: Loading of fluid elements.

The surface-based fluid cavity capability requires a surface to be defined on the boundary of the fluid cavity. The underlying elements on which the surface is defined can be solid elements, structural elements, or surface elements. In these tests the cavity boundary is represented with surface elements in the three-dimensional model. Since Abaqus does not provide two-dimensional surface elements, solid elements are used in the two-dimensional models to define the exterior surface. The solid elements are removed at the start of the first analysis step and are, therefore, not involved in the solution of the problem.

The two-dimensional fluid block measures 1 × 1 and has unit thickness, while the three-dimensional fluid block measures 1 × 1 × 1. Node 1 is the cavity reference node for the fluid cavity. In each case a single grounded spring acting in the *x*-direction is attached to a node on the outermost face of the model perpendicular to the *x*-direction. In addition, all nodes on this face are constrained to displace equally in the *x*-direction. The downward force is applied as a concentrated load to a single node on the uppermost face of the model perpendicular to the *y*-direction. All nodes on this face are constrained to displace equally in the *y*-direction. Finally, a grounded spring of negligible stiffness acting in the *y*-direction is attached to a single node on this face to preclude solver problems in the solution.

The axisymmetric fluid cylinder has a radius of 1 and a height of 1. Node 1 is the cavity reference node for the fluid cavity. In the first problem a single grounded spring acting in the *r*-direction is attached to a node on the outermost face of the model perpendicular to the *r*-direction. All nodes on this face are additionally constrained to displace equally in the *r*-direction. The downward force is applied as a concentrated load to a single node on the uppermost face of the model perpendicular to the *z*-direction. All nodes on this face are constrained to displace equally in the *z*-direction. Finally, a grounded spring of negligible stiffness acting in the *z*-direction is attached to a single node on this face to preclude solver problems in the solution. In the second problem all nodes are fixed in space, and the pressure inside the fluid is specified at node 1. No external force is specified, and no springs are used in the model.

Material: Fluid: incompressible, density = 10.0 (arbitrary).

Spring: $k =$ 400.

Loading: The concentrated force applied to all models except the second axisymmetric analysis ($F_y = -600$ at node 4) is ramped linearly from zero to the final value of −600 over a single static step. Results are reported at the end of the step.

$p =$ 1 for the second axisymmetric analysis.

Two-dimensional boundary conditions: $u_x =$ 0 at node 4; u_x is constrained to be equal at nodes 2 and 3.

$u_y =$ 0 at node 2; u_y is constrained to be equal at nodes 3 and 4.

Three-dimensional boundary conditions: $u_x =$ 0 at nodes 4, 5, and 8; u_x is constrained to be equal at nodes 2, 3, 6, and 7.

$u_y =$ 0 at nodes 2, 5, and 6; u_y is constrained to be equal at nodes 3, 4, 7, and 8.

$u_z =$ 0 at nodes 2 through 8.

Axisymmetric boundary conditions—Problem 1: $u_r =$ 0 at node 4; u_r is constrained to be equal at nodes 2 and 3.

$u_z =$ 0 at node 2; u_z is constrained to be equal at nodes 3 and 4.

Axisymmetric boundary conditions—Problem 2: $u_r =$ 0 at nodes 2, 3, and 4.

$u_z =$ 0 at nodes 2, 3, and 4.

Reference solution

Since the fluid is incompressible, the original fluid volume should be maintained. For the two-dimensional and three-dimensional cases CVOL = 1.0, and for the axisymmetric case CVOL = π.

For the second axisymmetric problem, the reaction forces at the nodes are as follows:

Node	RF_r	RF_z
2	$-\pi$	0.0
3	$-\pi$	$-2\pi/3$
4	0.0	$-\pi/3$

Results and discussion

Table 1: F2D2 results.

Node	u_x	u_y	PCAV	CVOL
1			376.9	1.000
2	0.5919	0.0		
3	0.5919	–0.3718		
4	0.0	–0.3718		

Table 2: F3D3 results.

Node	u_x	u_y	u_z	PCAV	CVOL
1				376.9	1.000
2	0.5919	0.0	0.0		
3	0.5919	–0.3718	0.0		
4	0.0	–0.3718	0.0		
5	0.0	0.0	0.0		
6	0.5919	0.0	0.0		
7	0.5919	–0.3718	0.0		
8	0.0	–0.3718	0.0		

Table 3: F3D4 results.

Node	u_x	u_y	u_z	PCAV	CVOL
1				376.9	1.000
2	0.5919	0.0	0.0		
3	0.5919	–0.3718	0.0		
4	0.0	–0.3718	0.0		
5	0.0	0.0	0.0		

Node	u_x	u_y	u_z	PCAV	CVOL
6	0.5919	0.0	0.0		
7	0.5919	–0.3718	0.0		
8	0.0	–0.3718	0.0		

Table 4: FAX2 results: Problem 1.

Node	u_r	u_z	PCAV	CVOL
1			88.25	3.142
2	0.4711	0.0		
3	0.4711	–0.5380		
4	0.0	–0.5380		

Table 5: FAX2 results: Problem 2.

Node	RF_r	RF_z	PCAV	CVOL
1			1.0	3.142
2	–3.1416	0.0		
3	–3.1416	–2.0944		
4	0.0	–1.0472		

Input files

ef22sxso.inp	F2D2 elements.
ef33sxso.inp	F3D3 elements.
ef34sxso.inp	F3D4 elements.
efa2sxso.inp	FAX2 elements, problem 1.
efa2sxsr.inp	FAX2 elements, problem 2.

Fluid link element

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Features tested

This section provides basic verification tests for the fluid link element that is generated in Abaqus/Standard when the fluid exchange capability is used to model flow between two fluid-filled cavities.

Connected fluid cavities

Elements tested

FLINK F2D2

Problem description

A fluid link element is created when the fluid exchange capability is used to transfer fluid between two vessels filled with incompressible fluid, as shown in *Figure 1*. One of the vessels is subjected to internal pressure by applying a load F. The other vessel is always maintained at zero pressure. The difference in pressures between the two vessels causes fluid to be transferred. Two analyses are performed to verify the fluid transfer rate between the two vessels by specifying the bulk viscosity and the mass flow rate as a function of pressure difference and temperature.

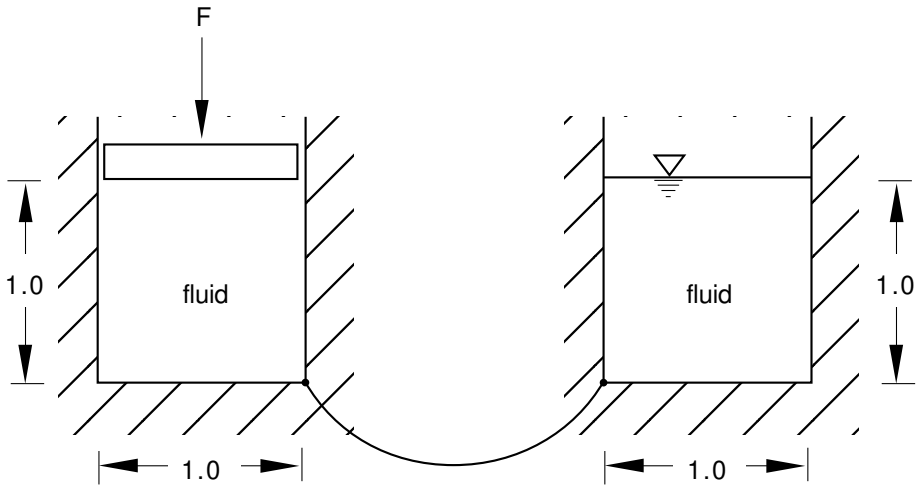


Figure 1: Fluid link analysis.

Each vessel is modeled using a two-dimensional fluid block that measures 1 × 1 with unit thickness, as shown in *Figure 2*. Nodes 1 and 11 are the cavity reference nodes for the two fluid cavities. The downward force on the first fluid cavity is applied as a concentrated load to node 4 in the y-direction. Nodes 3 and 4 are constrained to displace equally in the y-direction. Nodes 13 and 14 are also constrained to displace equally in the y-direction. Finally, grounded springs of very small stiffness acting in the y-direction are attached to nodes 4 and 14 to preclude solver problems in the solution.

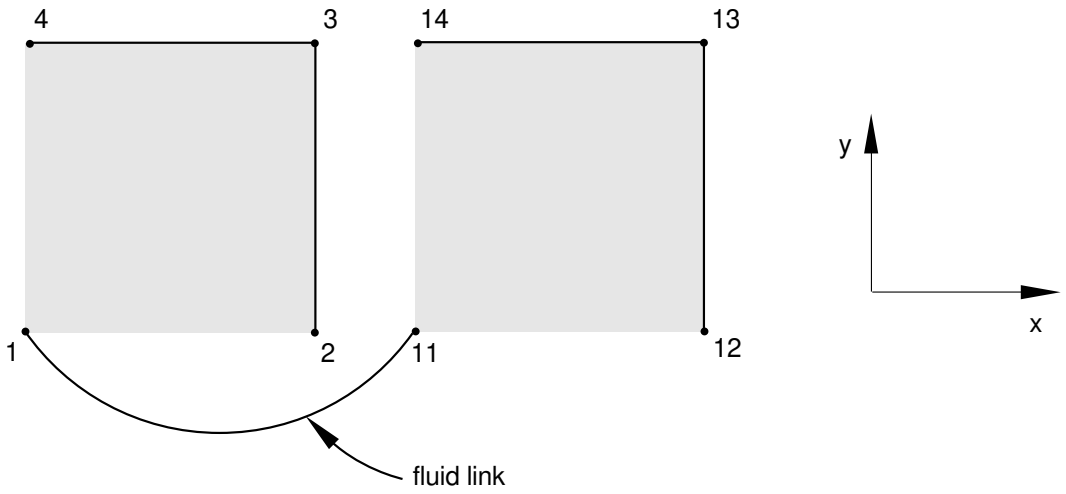


Figure 2: Fluid link model.

Material: Fluid: incompressible, density = 10.0 (arbitrary).

Fluid link:

Specifying the bulk viscosity			
C_V	C_H	$\bar{\theta}$	Field variable
10	0.0	10	1.0
1.0	0.0	100	1.0
10	0.001	10	2.0
1.0	0.001	100	2.0

The data used when specifying the mass flow rate as a function of pressure difference and temperature were computed using the implicit functional relationship between q and Δp discussed in *Fluid Exchange Definition*, and the values of C_V and C_H in the above table. To capture the nonlinear relationship between q and Δp accurately, 33 values of q were included in the data lines option for various combinations of $\bar{\theta}$ and the one field variable.

Loading: The concentrated force of 100 units is applied instantaneously over all static steps. In the first step the temperature and the field variable are held fixed at 10 and 1, respectively, for a time period of 0.20. In the second step the temperature is ramped from 10 to 100 for a time period of 0.01, while the field variable remains fixed at 1. The third step is a dummy perturbation step. This step is included to verify that an intermittent perturbation step has no effect on the subsequent general step. In the fourth step the temperature is held fixed at 100, with the field variable instantaneously changed to 2 for a time period of 0.01. Results are reported at the end of each general step.

Reference solution

Since the fluid is incompressible, the total fluid volume should be maintained; i.e., CVOL=2.0. The pressure in the first cavity should always be 100. Because of the presence of grounded springs of very small stiffness, the pressure in the second cavity is not zero.

Results and discussion

The results for the analyses compare quite well with one another. The agreement between the two models could be further improved by refining the tabular data for the mass flow rate model to better represent the nonlinear relationship between q and Δp as defined by the bulk viscosity model.

Specifying the bulk viscosity				
Step	$PCAV_1$	$CVOL_1$	$PCAV_{11}$	$CVOL_{11}$
1	100.0	0.800	2.00E−7	1.20
2	100.0	0.778	2.22E−7	1.22
4	100.0	0.683	3.17E−7	1.32

Specifying the mass flow rate as a function of pressure difference and temperature				
Step	$PCAV_1$	$CVOL_1$	$PCAV_{11}$	$CVOL_{11}$
1	100.0	0.800	2.00E−7	1.20
2	100.0	0.778	2.22E−7	1.22
4	100.0	0.683	3.18E−7	1.32

Input files

efl2sfsp.inp	TYPE=BULK VISCOSITY.
efl2stsp.inp	TYPE=MASS RATE LEAKAGE.

Single fluid cavity with a fluid link element

Elements tested

FLINK F2D2

Problem description

A fluid link element with one end connected and the other end free is used to transfer fluid to a single fluid cavity. The vessel is modeled using a two-dimensional fluid block that measures 1 × 1 with unit thickness. The model in this example is identical to the model shown in *Figure 2* except that the cavity defined by nodes 12, 13 and 14 is absent. Node 1 is the cavity reference node for the fluid cavity. Node 11 is connected to the fluid link element but not to a fluid cavity. Nodes 3 and 4 are constrained to displace equally in the y-direction. A grounded spring of unit stiffness acting in the y-direction is attached to node 4.

Two models are considered, one with an incompressible hydraulic fluid and the other with a compressible pneumatic fluid. The hydraulic fluid is given an arbitrary fluid density of $\rho = 10$. For the pneumatic fluid the molecular weight, MW, is set to 660; the universal gas constant, $\tilde{R}$, is set to 1; and the absolute zero temperature, θ_Z, is set to —460. See *Fluid Cavity Definition* for details. The fluid link is defined by specifying bulk viscosity as C_V=0.1 and C_H=0.

It is a simple exercise to show that with a single fluid link element and fixed temperature the change in mass in the fluid cavity is given by $\Delta m = \rho \Delta V + \rho_R V_R \left(\frac{\rho - \rho_R}{\rho_R} \right)$, where V_R is the initial volume of the fluid cavity

and ΔV is the change in volume of the fluid cavity with respect to V_R. For an incompressible hydraulic fluid $\rho = \rho_R$, in which case the change in mass is simply $\Delta m = \rho_R \Delta V$.

Loading: Four steps are used in the analyses. In Step 1 a constant mass flow rate of 10 is applied to node 11 on the fluid link element using the mass flow rate specified for a fluid cavity. In Step 2 the fluid flux loading is removed and the pressure at node 11 is held at its value at the end of step one using a boundary condition with the option of fixing a degree of freedom at its current value at the start of the step. In Step 3 the pressure at node 11 is ramped up to 5. Finally, in Step 4 all pressure boundary conditions are removed, and the system comes to rest.

Results and discussion

A comparison of the Abaqus results for the cavity pressure (pressure at node 1) to exact solutions for both the hydraulic and pneumatic fluids is shown in *Figure 3*. It is clear that Abaqus is accurately modeling the fluid cavity response.

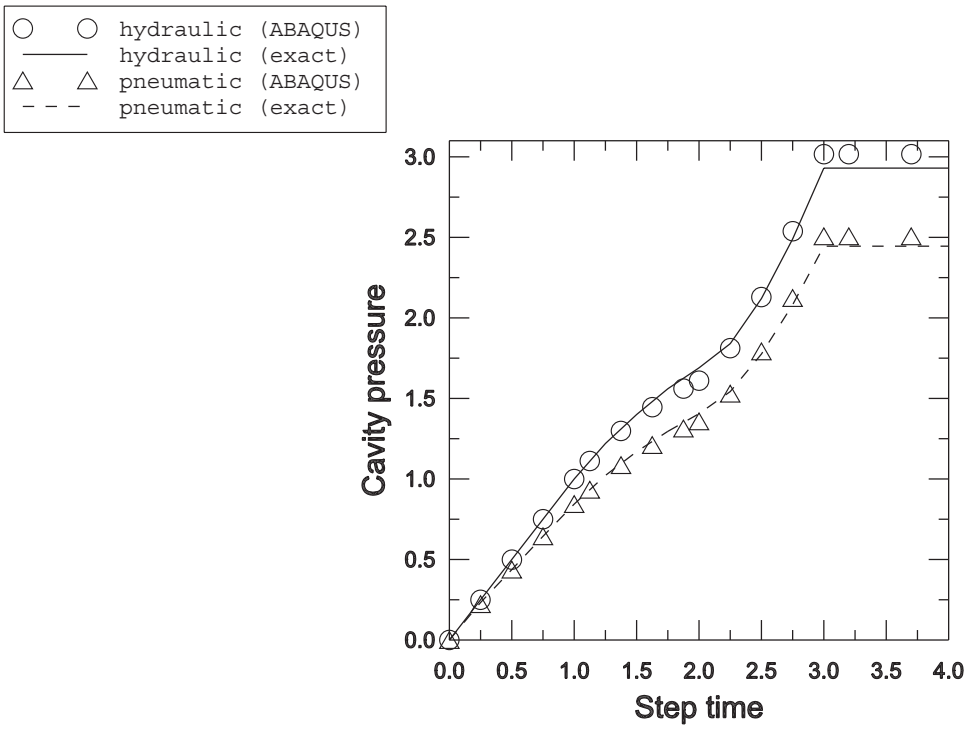


Figure 3: Cavity pressure.

Input files

onecav_hydr.inp	Hydraulic fluid.
onecav_pneu.inp	Pneumatic fluid.

Fluid pipe element

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Features tested

This section provides basic verification tests for the fluid pipe and fluid pipe connector elements, which are used to model fluid flow in a pipe in Abaqus/Standard.

Elements tested

FP2D2 FPC2D2 FP3D2 FPC3D2

Problem description

Fluid pipe elements are used to model the flow of fluid in a wellbore, as shown in *Figure 1*. The well consists of a vertical, a horizontal, and a deviated section. The deviated section is connected to the horizontal and vertical sections by fluid pipe connector elements. Two scenarios of laminar and turbulent flows are modeled in the wellbore. Point A is located 2000 m below ground level, and point D is located at ground level. The lengths of the horizontal and vertical sections are 1000 m and 500 m, respectively. The diameter of the pipe is 0.1 m. The connectors are modeled as two elbow joints. The connector loss coefficient used in this model is 0.64.

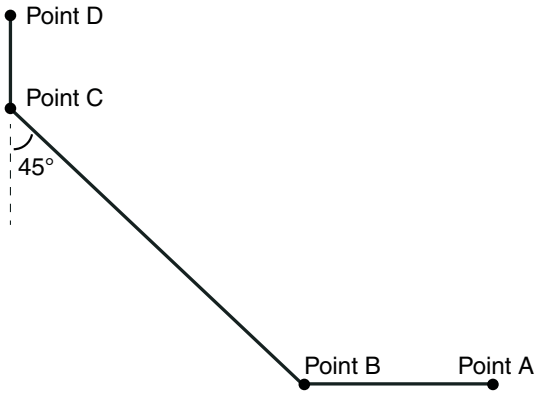


Figure 1: Wellbore geometry.

Material: The fluid is modeled as an incompressible liquid with a density of 1000 kg/m^3 and viscosity of 0.001 Pa·sec.

Loading: For laminar flow an inlet fluid flow rate of 12.4×10^{-3} m^3/s at point D and a pressure of 34.47 MPa at point A are prescribed. For turbulent flow an inlet flow rate of 10.77×10^{-3} m^3/s at point D and a pressure of 53.1 MPa at point A are specified.

Reference solution

Since the fluid is incompressible, the pressures can be computed using the Bernoulli equation for pipe flow.

Results and discussion

The analytical pressures computed using the Bernoulli equation for pipe flow at points A to D are shown in the table below. Since the prescribed boundary conditions for the laminar and turbulent flows are different, the computed pressures are different for the two cases. The computed pressures for the laminar and turbulent flows in the deviated wellbore match the analytical solution.

Point location	Laminar flow pressure (MPa)	Turbulent flow pressure (MPa)
A	34.47	53.1
B	34.47	52.95
C	19.77	37.92
D	14.87	32.94

Input files

wellbore_laminar_2d.inp	FP2D2 and FPC2D2 elements with laminar flow.
wellbore_laminar_3d.inp	FP3D2 and FPC3D2 elements with laminar flow.
wellbore_turbulent_2d.inp	FP2D2 and FPC2D2 elements with turbulent flow.
wellbore_turbulent_3d.inp	FP3D2 and FPC3D2 elements with turbulent flow.

Thermal fluid pipe elements

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Features tested

This section provides basic verification tests for the thermal fluid pipe and annular thermal fluid pipe elements used to model fluid flow in a pipe in Abaqus/Standard.

Elements tested

FP3D2AT FP3D2T

Problem description

Thermal fluid pipe elements are used to model fluid flow in a vertical wellbore. Two scenarios of laminar and turbulent flows are modeled in the wellbore. Point A is located 2000 m below ground level, and point B is located at ground level. The diameter of the pipe is 0.1 m. Gravity effects are not included in this model. Four different viscosity definitions (Newtonian, Herschel-Bulkley, Bingham plastic, and power law) are used to verify that the computed pressures match the analytical solution. The temperature evolution due to convective effects at the fluid–pipe wall interface is modeled using either a constant temperature approach or a constant flux approach.

Material: The fluid is modeled as an incompressible liquid with a density of 1000 kg/m^3:

- Newtonian fluid: viscosity defined to be 0.001 Pa·sec
- Herschel-Bulkley fluid: shear viscosity at low shear rates of 1×10^{-5} Pa·sec, a yield shear stress of 1×10^{-3} Pa, a consistency index of 0.001, and a flow behavior index of 0.8
- Bingham plastic fluid: shear viscosity at the low shear rate of 1×10^{-3} Pa·sec, a yield shear stress of 0.01 Pa, and a consistency index of 0.001
- Power law fluid: consistency index of 0.001, a flow behavior index of 0.8, a maximum viscosity of 1×10^{7} Pa·sec, and a minimum viscosity of 1×10^{-6} Pa·sec

Loading: For laminar flow, an inlet fluid flow rate of 5×10^{-4} m^3/s at point B and a pressure of 0 MPa at point A are prescribed. For turbulent flow, an inlet flow rate of 1.5×10^{-3} m^3/s at point B and a pressure of 0 MPa at point A are specified. For the thermal solution, the fluid temperature is prescribed to be 100 and the outer wall temperature is prescribed to be 10.

Reference solution

Since the fluid is incompressible, the pressures can be computed using the Bernoulli equation for pipe flow.

Results and discussion

The analytical pressures computed using the Bernoulli equation for pipe flow at point B are shown in the table below. Since the prescribed boundary conditions for the laminar and turbulent flows are different, the computed

pressures are different for the two cases. The computed pressures for the laminar and turbulent flows in the vertical wellbore match the analytical solution.

	Pressure drop in Pa			
Viscosity definition	Analytical—laminar flow	Abaqus/Standard—laminar flow	Analytical—turbulent flow	Abaqus/Standard—turbulent flow
Herschel-Bulkley	1241.8	1241.8	4943.7	4941
Bingham plastic	1325.9	1325.9	4509.6	4512
Power law	85.22	85.21	326.6	325.8
Newtonian	81.48	81.48	3267	3259

Viscosity definition	**Analytical temperature °C at inner wall—constant temperature-laminar flow**	**Abaqus/Standard temperature °C at inner wall—constant temperature-laminar flow**
Herschel-Bulkley	60.3	60.3
Bingham plastic	60.3	60.3
Power law	60.3	60.3
Newtonian	60.3	60.3

Viscosity definition	**Analytical temperature °C at inner wall—constant flux- laminar flow**	**Abaqus/Standard temperature °C at inner wall—constant flux-laminar flow**
Herschel-Bulkley	69.3	69.3
Bingham plastic	67.8	67.8
Power law	64.7	64.7
Newtonian	64.2	64.2

Input files

wellbore_newtonian_laminar_constFlux.inp	FP3D2T element with Newtonian viscosity definition, laminar flow, and constant flux approach to determine Nusselt number for convection.
wellbore_newtonian_laminar_constTemp.inp	FP3D2T element with Newtonian viscosity definition, laminar flow, and constant temperature approach to determine Nusselt number for convection.
wellbore_BP_laminar_constFlux.inp	FP3D2T element with Bingham plastic viscosity definition, laminar flow, and constant flux approach to determine Nusselt number for convection.
wellbore_BP_laminar_constTemp.inp	FP3D2T element with Bingham plastic viscosity definition, laminar flow, and constant temperature approach to determine Nusselt number for convection.
wellbore_HB_laminar_constFlux.inp	FP3D2T element with Herschel-Bulkley viscosity definition, laminar flow, and constant flux approach to determine Nusselt number for convection.
wellbore_HB_laminar_constTemp.inp	FP3D2T element with Herschel-Bulkley viscosity definition, laminar flow, and constant temperature approach to determine Nusselt number for convection.

wellbore_power_laminar_constFlux.inp	FP3D2T element with Herschel-Bulkley viscosity definition, laminar flow, and constant flux approach to determine Nusselt number for convection.
wellbore_power_laminar_constTemp.inp	FP3D2T element with Herschel-Bulkley viscosity definition, laminar flow, and constant temperature approach to determine Nusselt number for convection.
wellbore_annulus_newtonian_laminar.inp	FP3D2AT element with Newtonian viscosity definition, laminar flow.
wellbore_annulus_HB_laminar.inp	FP3D2AT element with Herschel-Bulkley viscosity definition, laminar flow.
wellbore_annulus_BP_laminar.inp	FP3D2AT element with Bingham plastic viscosity definition, laminar flow.
wellbore_annulus_power_laminar.inp	FP3D2AT element with power law viscosity definition, laminar flow.
wellbore_newtonian_turbulent.inp	FP3D2T element with Newtonian viscosity definition, turbulent flow.
wellbore_HB_turbulent.inp	FP3D2AT element with Herschel-Bulkley viscosity definition, turbulent flow.
wellbore_BP_turbulent.inp	FP3D2T element with Bingham plastic viscosity definition, turbulent flow.
wellbore_power_turbulent.inp	FP3D2AT element with power law viscosity definition, turbulent flow.

Temperature-dependent film condition

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

CPE3T CPE4RT CPS3T CPS4RT DC2D4 S4RT

Features tested

Temperature-dependent film conditions.

Problem description

An infinite plate of width 0.1 unit and thickness 1 unit is considered. A zero flux boundary condition is imposed on all of the surfaces except the top surface. A film condition and sink temperature are imposed on the top surface, and the transient solution to the heat transfer problem is sought.

In Abaqus/Standard the problem is modeled with 10 DC2D4 elements of dimension 0.01×0.01. In Abaqus/Explicit two-dimensional (plane strain and plane stress) elements are used to model the plate: 10 elements are used through the width of the plate when using CPE4RT and CPS4RT elements, while 20 elements are used when using CPE3T and CPS3T elements. The problem is also modeled using S4RT elements in Abaqus/Explicit. Only one coupled thermal shell element is used, and the shell's thickness represents the length of the model. The film condition is applied on one face of the shell, and a large number of temperature points are considered through the thickness (19 points, which is the maximum allowable temperature points.)

Material: Thermal conductivity, $k = 1.4$; sink temperature, $\theta_0 = 100(1 + t/3600)$; specific heat, $c_p = 260$; film coefficient, $h = 10 + 0.02\theta$; density, $\rho = 7800$; initial temperature, $\theta_i = 0$.

In Abaqus/Explicit dummy mechanical properties are prescribed to complete the material definition.

Results and discussion

The transient solution at $t = 3600$ units is plotted for all four cases; the finite difference solution is plotted as a solid line, and the three finite element results as markers at the centroid of the elements.

The results obtained with Abaqus/Explicit are in close agreement with those obtained with Abaqus/Standard.

Input files

Abaqus/Standard input files

ec24dfd1.f — Fortran program to compute the finite difference solution to the differential equation

$$k\frac{\partial^2 \theta}{\partial x_2^2} = \rho c_p \frac{\partial \theta}{\partial t}$$

	with appropriate boundary conditions for the film and sink conditions. The solution is computed at 101 points through the width of the plate at time steps of 0.01 units.
ec24dfd2.inp	Finite element model of the problem as described above.
ec24dfd3.inp	Finite element model with temperature dependent film condition prescribed through user subroutine *FILM*.
ec24dfd3.f	User subroutine *FILM* used in ec24dfd3.inp.
ec24dfd4.inp	Finite element model where the film condition is changed using a field variable which is prescribed through user subroutine *UFIELD*.
ec24dfd4.f	User subroutine *UFIELD* used in ec24dfd4.inp.
ec24dfd5.inp	Same as problem ec24dfd2.inp with surface-based loads.
ec24dfd6.inp	Same as problem ec24dfd3.inp with surface-based loads.
ec24dfd6.f	User subroutine *FILM* used in ec24dfd6.inp.
ec24dfd7.inp	Same as problem ec24dfd4.inp with surface-based loads.
ec24dfd7.f	User subroutine *UFIELD* used in ec24dfd7.inp.

Abaqus/Explicit input files

tempdepfilm_xpl_cpe3t.inp	CPE3T elements.
tempdepfilm_xpl_cpe4rt.inp	CPE4RT elements.
tempdepfilm_xpl_cps3t.inp	CPS3T elements.
tempdepfilm_xpl_cps4rt.inp	CPS4RT elements.
tempdepfilm_xpl_s4rt.inp	S4RT elements.

Figures

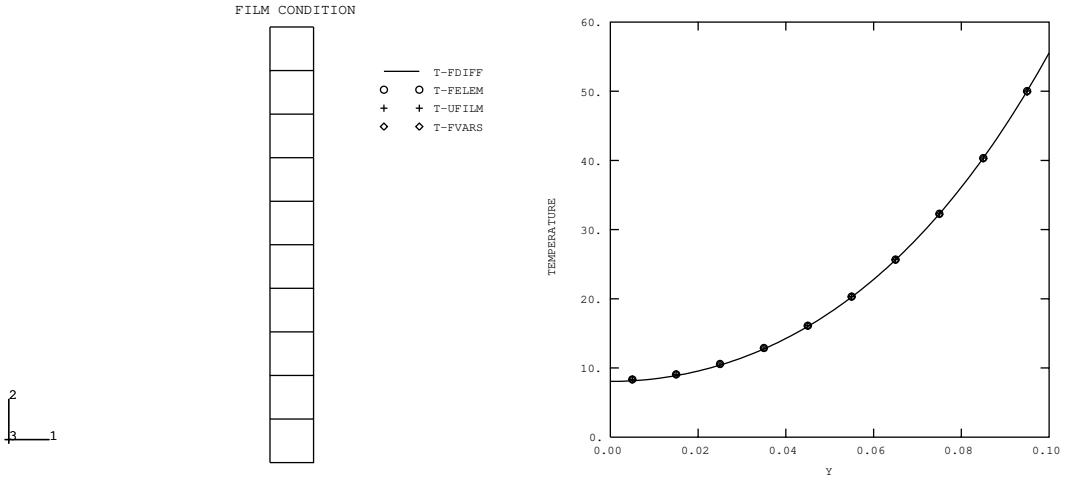


Figure 1: Finite element model and temperature profile (Abaqus/Standard).

Surface-based pressure penetration

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D4 C3D6 C3D8 C3D10 C3D10M C3D15 C3D20
CAX4 CAX8
CCL9 CCL12 CCL18 CCL24
CPE4 CPE8
M3D4
S4 SC6R SC8R SAX1 SAX2

Features tested

Contact between a deformable body and a rigid surface and contact between two deformable surfaces exposed to a fluid pressure at either one or both ends of the surfaces are tested through the use of pressure penetration loads.

Problem description

We consider fluid pressure penetration loading for the six model configurations shown in *Figure 1* through *Figure 6*. Fluid pressure penetration loading is such that the eventual extent of contact surface regions exposed to fluid pressure are determined during a simulation as contact conditions change.

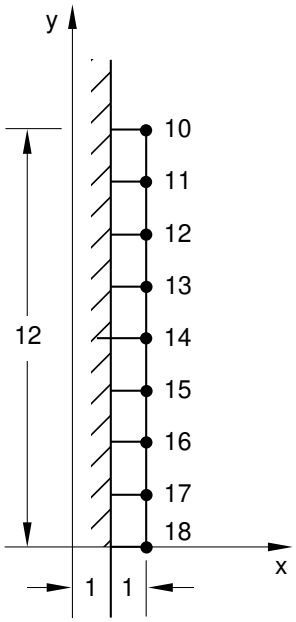


Figure 1: Contact between a deformable flat surface and a rigid flat surface in two dimensions.

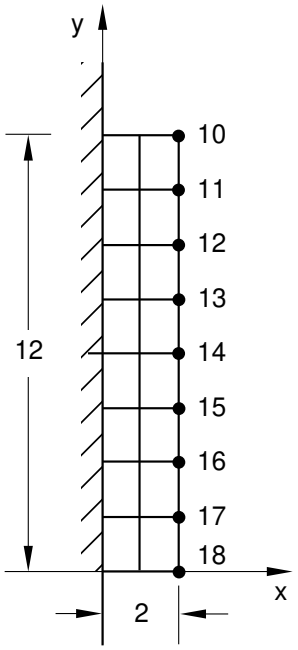


Figure 2: Contact between two deformable flat surfaces with matching meshes in two dimensions.

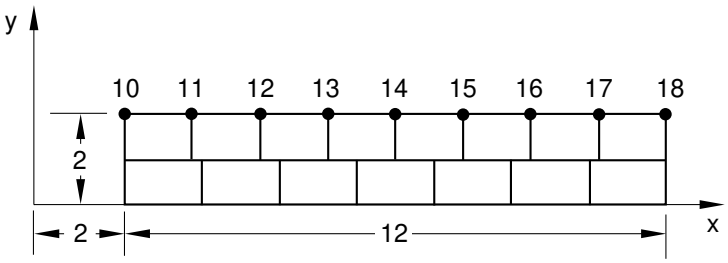


Figure 3: Contact between two deformable flat surfaces with nonmatching meshes in two dimensions.

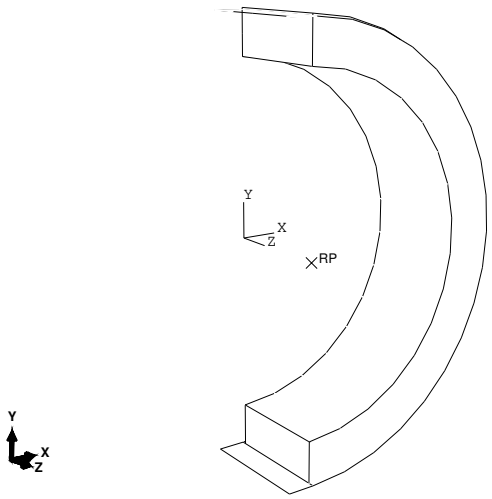


Figure 4: Contact between deformable and rigid cylindrical surfaces in three dimensions.

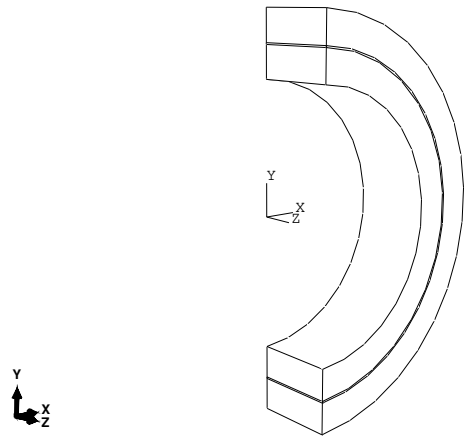


Figure 5: Contact between two deformable cylindrical surfaces in three dimensions.

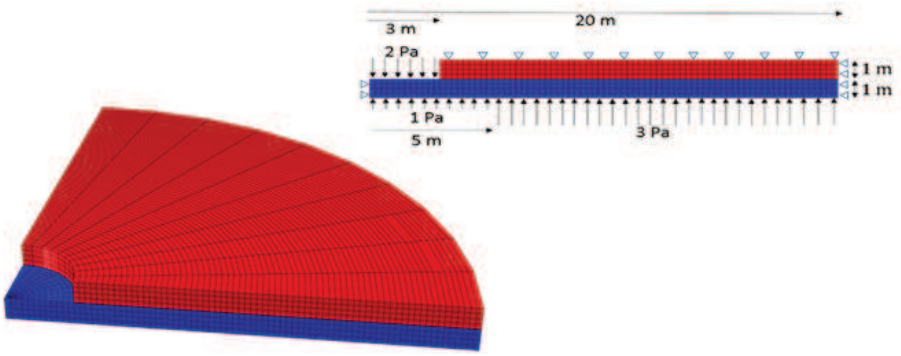


Figure 6: Contact between two flat deformable plates with a hole in the center of one plate.

Model: Brief descriptions of the model configurations follow:

- *Figure 1* shows a two-dimensional model with an interface between a deformable flat surface and a rigid flat surface.
- *Figure 2* shows a two-dimensional model with an interface between two deformable flat surfaces with matching meshes.
- *Figure 3* shows a two-dimensional model with an interface between two deformable flat surfaces with nonmatching meshes.
- *Figure 4* shows a three-dimensional model with an interface between deformable and analytical cylindrical rigid surfaces.
- *Figure 5* shows a three-dimensional model with an interface between two deformable cylindrical surfaces.
- *Figure 6* shows a three-dimensional model with an interface between two flat deformable plates with a hole in the center of one plate.

Material: All deformable bodies in the models have the following properties:

Young's modulus = 1×10^5; Poisson's ratio = 0.3.

Boundary conditions: For the models illustrated in *Figure 1*, *Figure 2*, and *Figure 3*, the rigid surface is constrained in all degrees of freedom. When the contact between two deformable surfaces is considered, the nodes at $x = 0$ in *Figure 2* and the nodes at $y = 0$ in *Figure 3* are constrained in all degrees of freedom.

For the models illustrated in *Figure 4* and *Figure 5*, the rigid surface is constrained in all degrees of freedom. The nodes at the two ends of the rings are constrained in the first degree of freedom, and the nodes at the back surfaces are constrained in the third degree of freedom; when the contact between two deformable surfaces is considered, the inner and outer ring surfaces of the model are also constrained in all degrees of freedom.

For the model illustrated in *Figure 6*, the nodes on the top of the top plate and the nodes on the sides of both the top and bottom plates are constrained in all degrees of freedom.

Loading: For the models illustrated in *Figure 1* and *Figure 2*:

Step 1:

A nonuniform displacement, $u = \frac{u_{14}}{1.25^d}$, is applied in the negative *x*-direction on the surface of $x = 2.0$ when solid elements are used. u_{14} is the displacement at node 14, and *d* is the distance measured from node 14. When shell elements are used, a nonuniform pressure, $P = P_{14}\left(1 - \frac{dc-0.75}{9}\right)$, is applied on the surface of $x = 2.0$. P_{14} is the pressure at the element containing node 14, and dc is the distance measured from node 14 to the center of an element.

Step 2:

Two ends of the contacting surfaces ($y = 0$ and $y = 12.0$) are exposed to a fluid pressure with a magnitude of 800.0.

Step 3:

Increase the fluid pressure to 900.0 at both ends.

For the model illustrated in *Figure 3*:

Step 1:

A nonuniform displacement, $u = \frac{u_{14}}{1.25^d}$, is applied in the negative *y*-direction on the surface of $y = 2.0$.

Step 2:

One end of the contacting surfaces, $x = 2$, is exposed to a fluid pressure with a magnitude of 550; and the other end, $x = 14$, is exposed to a fluid pressure with a magnitude of 800.

Step 3:

Increase the magnitudes of the fluid pressure to 650 from 550 and to 900 from 800, respectively, at both ends.

For the models illustrated in *Figure 4* and *Figure 5*:

Step 1:

The models consist of two parts with contact surfaces initially overclosed. This initial overclosure is resolved automatically in the step by using the automatic shrink fit capability in the contact interference definition.

Step 2:

There are two places initially exposed to a fluid pressure with a magnitude of 20000: at the outside corner of the contact surfaces at the bottom end of the rings and along the whole edges of the contact surfaces at the upper end of the rings.

For the model illustrated in *Figure 6*:

Step 1:

A nonuniform pressure is applied to the bottom surface of the bottom plate, as shown.

Step 2:

Fluid is initially exposed at the left end of the bottom plate, $x = 0$ and $y = 1$, and the left end of the top plate, $x = 3$ and $y = 1$. The fluid becomes exposed across the hole before being impeded by the contact between the bottom surface of the top plate and the top surface of the bottom plate.

Results and discussion

The contact pressure and the fluid pressure at each secondary node on the contacting surfaces are output. *Figure 7* specifically shows the deformation of the three-dimensional model in the middle of step 2, and *Figure 8* shows the fluid expansion of the axisymmetric model at the end of step 2 both with and without cohesive behavior defined.

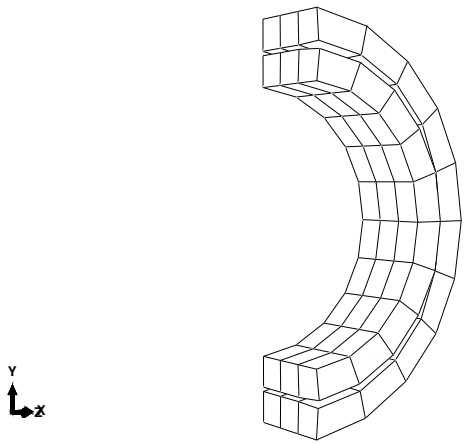


Figure 7: Deformation of the model with two deformable surfaces in three dimensions.

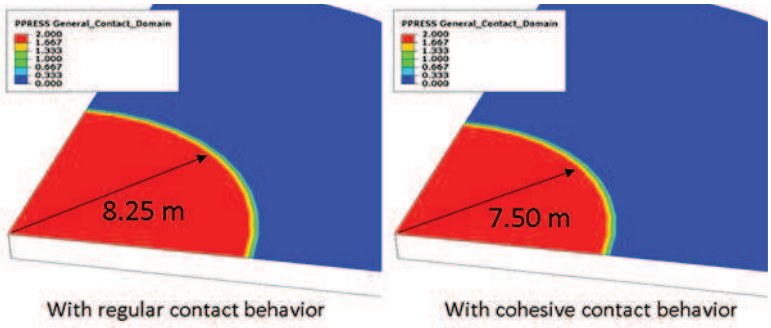


Figure 8: Fluid pressure PPRESS on the bottom plate at the end of the analysis with and without cohesive behavior.

Input files

Two flat surfaces in contact as illustrated by *Figure 1*:

ei22srs1_ppen.inp	CPE4 elements using contact pairs in Abaqus/Standard.
ei22srs1_ppendsl_gc.inp	CPE4 elements using general contact in Abaqus/Standard.
ei23srs1_ppen.inp	CPE8 elements using contact pairs in Abaqus/Standard.
eia2srs1_ppen.inp	CAX4 elements using contact pairs in Abaqus/Standard.
eia2srs3_ppen.inp	CAX4 elements with a rigid body created from MAX1 elements using contact pairs in Abaqus/Standard.
eia3srs1_ppen.inp	CAX8 elements using contact pairs in Abaqus/Standard.
eia3srs1_ppen_auglagr.inp	CAX8 elements using contact pairs in Abaqus/Standard.
eia3srs3_ppen.inp	CAX8 elements with a rigid body created from MAX1 elements using contact pairs in Abaqus/Standard.
eia3srs3_ppen_auglagr.inp	CAX8 elements with a rigid body created from MAX1 elements using contact pairs in Abaqus/Standard.
eia2srs2_ppen.inp	SAX1 elements using contact pairs in Abaqus/Standard.
eia2srs4_ppen.inp	SAX1 elements with a rigid body created from SAX1 elements using contact pairs in Abaqus/Standard.
eia3srs2_ppen.inp	SAX2 elements using contact pairs in Abaqus/Standard.
eia3srs4_ppen.inp	SAX2 elements with a rigid body created from SAX1 elements using contact pairs in Abaqus/Standard.

Two flat surfaces in contact as illustrated by *Figure 2*:

ei22sss1_ppen.inp	CPE4 elements using contact pairs in Abaqus/Standard.
ei23sss1_ppen.inp	CPE8 elements using contact pairs in Abaqus/Standard.
ei23sss1_ppen_auglagr.inp	CPE8 elements using contact pairs in Abaqus/Standard.
eia2sss1_ppen.inp	CAX4 elements using contact pairs in Abaqus/Standard.
eia2sss2_ppen.inp	SAX1 elements using contact pairs in Abaqus/Standard.
eia2sss3_ppen.inp	SAX1 and CAX4 elements using contact pairs in Abaqus/Standard.

Two flat surfaces in contact as illustrated by *Figure 3*:

ei22sss2_ppen.inp	CPE4 elements using contact pairs in Abaqus/Standard.
eia2sss4_ppen.inp	CAX4 elements using contact pairs in Abaqus/Standard.

Two cylindrical surfaces in contact as illustrated by *Figure 4*:

ver-ppen-c3d4-rigid.inp	C3D4 elements with an analytical rigid surface using contact pairs in Abaqus/Standard.
ver-ppen-c3d6-rigid.inp	C3D6 elements with an analytical rigid surface using contact pairs in Abaqus/Standard.
ver-ppen-c3d8-rigid.inp	C3D8 elements with an analytical rigid surface using contact pairs in Abaqus/Standard.
ver-ppen-c3d10-rigid.inp	C3D10 elements with an analytical rigid surface using contact pairs in Abaqus/Standard.
ver-ppen-c3d10m-rigid.inp	C3D10M elements with an analytical rigid surface using contact pairs in Abaqus/Standard.
ver-ppen-c3d15-rigid.inp	C3D15 elements with an analytical rigid surface using contact pairs in Abaqus/Standard.
ver-ppen-c3d20-rigid.inp	C3D20 elements with an analytical rigid surface using contact pairs in Abaqus/Standard.
ver-ppen-ccl9-rigid.inp	CCL9 elements with an analytical rigid surface using contact pairs in Abaqus/Standard.
ver-ppen-ccl12-rigid.inp	CCL12 elements with an analytical rigid surface using contact pairs in Abaqus/Standard.
ver-ppen-ccl18-rigid.inp	CCL18 elements with an analytical rigid surface using contact pairs in Abaqus/Standard.
ver-ppen-ccl24-rigid.inp	CCL24 elements with an analytical rigid surface using contact pairs in Abaqus/Standard.
ver-ppen-s4-rigid.inp	S4 elements with an analytical rigid surface using contact pairs in Abaqus/Standard.
ver-ppen-sc6r-rigid.inp	SC6R elements with an analytical rigid surface using contact pairs in Abaqus/Standard.
ver-ppen-sc8r-rigid.inp	SC8R elements with an analytical rigid surface using contact pairs in Abaqus/Standard.
ver-ppen-m3d4-rigid.inp	M3D4 elements with an analytical rigid surface using contact pairs in Abaqus/Standard.

Two cylindrical surfaces in contact as illustrated by *Figure 5*:

ver-ppen-c3d4.inp	C3D4 elements using contact pairs in Abaqus/Standard.
ver-ppen-c3d4_xpl.inp	C3D4 elements in Abaqus/Explicit.
ver-ppen-c3d4-mismatch.inp	C3D4 elements using contact pairs in Abaqus/Standard.
ver-ppen-c3d4-mismatch_gc.inp	C3D4 elements using general contact in Abaqus/Standard.
ver-ppen-c3d6.inp	C3D6 elements using contact pairs in Abaqus/Standard.
ver-ppen-c3d6_xpl.inp	C3D6 elements in Abaqus/Explicit.
ver-ppen-c3d8.inp	C3D8 elements using contact pairs in Abaqus/Standard.
ver-ppen-c3d8_xpl.inp	C3D8 elements in Abaqus/Explicit.
ver-ppen-c3d8-mismatch.inp	C3D8 elements using contact pairs in Abaqus/Standard.
ver-ppen-c3d8-mismatch_gc.inp	C3D8 elements using general contact in Abaqus/Standard.
ver-ppen-c3d10.inp	C3D10 elements using contact pairs in Abaqus/Standard.
ver-ppen-c3d10_gc.inp	C3D10 elements using general contact in Abaqus/Standard.
ver-ppen-c3d10_xpl.inp	C3D10 elements in Abaqus/Explicit.

ver-ppen-c3d10m.inp	C3D10M elements using contact pairs in Abaqus/Standard.
ver-ppen-c3d10m_gc.inp	C3D10M elements using general contact in Abaqus/Standard.
ver-ppen-c3d10m_xpl.inp	C3D10M elements in Abaqus/Explicit.
ver-ppen-c3d15.inp	C3D15 elements using contact pairs in Abaqus/Standard.
ver-ppen-c3d15_gc.inp	C3D15 elements using general contact in Abaqus/Standard.
ver-ppen-c3d20.inp	C3D20 elements using contact pairs in Abaqus/Standard.
ver-ppen-c3d20_gc.inp	C3D20 elements in using general contact Abaqus/Standard.
ver-ppen-cax4r_xpl.inp	CAX4R elements in Abaqus/Explicit.
ver-ppen-ccl9.inp	CCL9 elements using contact pairs in Abaqus/Standard.
ver-ppen-ccl12.inp	CCL12 elements using contact pairs in Abaqus/Standard.
ver-ppen-ccl18.inp	CCL18 elements using contact pairs in Abaqus/Standard.
ver-ppen-ccl24.inp	CCL24 elements using contact pairs in Abaqus/Standard.
ver-ppen-cpe4r_xpl.inp	CPE4R elements in Abaqus/Explicit.
ver-ppen-s4.inp	S4 elements using contact pairs in Abaqus/Standard.
ver-ppen-s4_xpl.inp	S4 elements in Abaqus/Explicit.
ver-ppen-sc6r.inp	SC6R elements using contact pairs in Abaqus/Standard.
ver-ppen-sc6r_gc.inp	SC6R elements using general contact in Abaqus/Standard.
ver-ppen-sc6r_xpl.inp	SC6R elements in Abaqus/Explicit.
ver-ppen-sc8r.inp	SC8R elements using contact pairs in Abaqus/Standard.
ver-ppen-sc8r_gc.inp	SC8R elements using general contact in Abaqus/Standard.
ver-ppen-sc8r_xpl.inp	SC8R elements in Abaqus/Explicit.
ver-ppen-m3d4.inp	M3D4 elements using contact pairs in Abaqus/Standard.
ver-ppen-m3d4_xpl.inp	M3D4 elements in Abaqus/Explicit.

Two flat surface in contact with a hole in one surface as illustrated by *Figure 6*:

ppen_2plates_xpl.inp	CAX4R elements in Abaqus/Explicit.
ppen_2plates_coh_xpl.inp	CAX4R elements with cohesive behavior in Abaqus/Explicit.

Gasket behavior verification

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

GK2D2 GK2D2N
GK3D2 GK3D2N GK3D4L GK3D4LN GK3D6 GK3D6L GK3D6LN GK3D6N GK3D6T GK3D6TN GK3D8 GK3D8N GK3D8T GK3D8TN GK3D12M GK3D12MN GK3D18 GK3D18N

GKAX4 GKAX4N GKAX6 GKAX6N
GKPE4 GKPE6 GKPS4 GKPS4N GKPS6 GKPS6N

Problem description

Gasket elements are used to model sealing components between structural members. They are designed primarily to provide appropriate pressure-closure behavior in the thickness direction, which is uncoupled from the transverse shear and membrane behavior. This uncoupled pressure-closure behavior is specified with the suboptions of the gasket behavior definition. These gasket behavior models are separate from the models in the material library. For some gasket behaviors that are not addressed readily by these special behavior models, such as coupled compression-membrane behaviors or through-thickness tensile behavior, Abaqus provides an alternative way for the user to model such behavior by specifying either a built-in or user-defined material model.

Gasket elements with all displacement degrees of freedom active at their nodes are used to model all three types of behavior (thickness-direction, membrane, and transverse shear). Elements that have only one displacement degree of freedom at their nodes consider the thickness-direction behavior only. Analyses are performed to verify that the generalized strains (displacements and strains) in the gaskets are obtained properly from the nodal displacements and that the generalized stresses (forces, forces per unit length, or stresses) are obtained properly from the generalized strains through the constitutive relations for the different types of behavior.

Behavior

The element tests included in this section cover three different types of model behavior in the thickness direction: elastic with damage, elastic-plastic with high initial stiffness so that yield occurs at the second data point given along the initial loading curve, and elastic-plastic with low initial stiffness so that initial yield occurs further along the initial loading curve. Rate-dependent (creep) effects through the thickness of the gasket may be added to the elastic-plastic model. These models are used with or without any initial gap. Thermal expansion is also considered along the thickness direction of the gasket with or without an initial void. The thickness-direction behavior is defined in terms of stress in most cases but is defined as force or force per unit length whenever it is appropriate for the element type.

Membrane behavior and transverse shear behavior are modeled as linear elastic. Thermal expansion is also considered for the membrane response.

Some tests involving viscoelastic effects (in the frequency domain) in conjunction with the elastic or elastic-damage models have also been included. These tests model the frequency-dependent stiffness and damping characteristics of gaskets for different levels (as measured by closure) of preload. Only thickness-direction viscoelastic behavior is modeled in these cases.

Field expansion is also tested for a 2–node two-dimensional gasket.

Model

Each model contains a set of 1 to 4 gasket elements of the same type. The initial geometry may or may not be aligned with the global axes. When gasket elements with thickness-direction-only behavior are used, the model may also contain a rigid element that is used to control the loading of the gasket. The tests for the gasket elements using suboptions of the material model are elastic-plastic creep analyses. Corresponding solid elements are included in the tests to facilitate the comparison of solutions.

The material properties for the tests involving viscoelastic behavior include specification of storage and loss moduli as functions of excitation frequency and level of preload (closure).

Loading history

The different tests include compression along the thickness direction of the gasket, shearing of the top surface of the gasket with respect to its bottom surface (whenever possible), and uniform extension or compression of the gasket membrane. The tests are displacement- or load-controlled through direct control of the gasket element's nodes or through a contact pair that involves the gasket and a rigid component compressing the gasket. The tests also include thermal loading in the gasket membrane and/or gasket thickness direction.

The tests performed along the thickness direction of the gasket involve, in most cases, a series of loading and unloading steps to verify that the user-prescribed loading and unloading curves are followed properly and that interpolation between user-specified unloading curves is done correctly.

The tests involving the modeling of viscoelastic behavior consist of steady-state harmonic oscillations at different excitation frequencies about base states that involve different levels of closure. These tests are displacement controlled.

Results and discussion

The results are obtained at the end of each step in terms of generalized stresses, generalized strains, reaction forces, and nodal displacements. The results obtained in each test match the results obtained by hand calculations.

Input files

ege4gcxx.inp GKPE4 elements, quasi-static analysis with creep.

Linear perturbation analysis:

eg22nlxp.inp GK2D2N elements, perturbation step with **LOAD CASE*.

egs4gdxp.inp GKPS4 elements, perturbation step with **LOAD CASE*.

Generation of missing nodes:

eg3rgdxm.inp GK3D18 elements.

eg3rndxm.inp GK3D18N elements, elastic with damage.

eg3rnhxm.inp GK3D18N elements, elastic-plastic (high initial stiffness).

Dependency on field variable and temperature:

ega4glxv.inp GKAX4 elements.

ega6glxv.inp GKAX6 elements.

ega6nhxv.inp GKAX6N elements.

egl4glxv.inp GK3D4L elements.

egl4nhxv.inp GK3D4LN elements.

egl6ghxv.inp	GK3D6L elements.
egl6nhxv.inp	GK3D6LN elements.

Solid element numbering:

eg3agdxs.inp	GK3D12M elements.
eg3rgdxs.inp	GK3D18 elements.
egs4gdxs.inp	GKPS4 elements.
egs6gdxs.inp	GKPS6 elements.

User subroutine *UVARM*:

ega4glxu.inp	GKAX4 elements.
ega4glxu.f	User subroutine *UVARM* used in ega4glxu.inp.
ega6nhxu.inp	GKAX6N elements.
ega6nhxu.f	User subroutine *UVARM* used in ega6nhxu.inp.
egl4glxu.inp	GK3D4L elements.
egl4glxu.f	User subroutine *UVARM* used in egl4glxu.inp.
egl6ghxu.inp	GK3D6L elements.
egl6ghxu.f	User subroutine *UVARM* used in egl6ghxu.inp.

Yield option:

ega4glxy.inp	GKAX4 elements.
ega4nhxy.inp	GKAX4N elements.
ega6glxy.inp	GKAX6 elements.
ega6nhxy.inp	GKAX6N elements.
egl4glxy.inp	GK3D4L elements.
egl4nhxy.inp	GK3D4LN elements.
egl6ghxy.inp	GK3D6L elements.
egl6nhxy.inp	GK3D6LN elements.

Slope drop option:

eg22gcxz.inp	GK2D2 elements, elastic-plastic with creep.
eg22gdxz.inp	GK2D2 elements, elastic with damage.
eg36gcxz.inp	GK3D6 elements, elastic-plastic with creep.
eg36gdxz.inp	GK3D6 elements, elastic with damage.
eg36gcxz_ctd.inp	GK3D6T elements, elastic-plastic with creep.
eg36gdxz_ctd.inp	GK3D6T elements, elastic with damage.
ega2gcxz.inp	GKAX2 elements, elastic-plastic with creep.
ega4glxz.inp	GKAX4 elements, elastic-plastic (low initial stiffness).
ega4nhxz.inp	GKAX4N elements, elastic-plastic (high initial stiffness).
ega6glxz.inp	GKAX6 elements, elastic-plastic (low initial stiffness).
ega6nhxz.inp	GKAX6N elements, elastic-plastic (high initial stiffness).
ege4gdxz.inp	GKPE4 elements, elastic with damage.
egl4gcxz.inp	GK3D4L elements, elastic-plastic with creep.
egl4gdxz.inp	GK3D4L elements, elastic with damage.
egl4glxz.inp	GK3D4L elements, elastic-plastic (low initial stiffness).

egl4nhxz.inp	GK3D4LN elements, elastic-plastic (high initial stiffness).
egl6ghxz.inp	GK3D6L elements, elastic-plastic (high initial stiffness).
egl6nhxz.inp	GK3D6LN elements, elastic-plastic (high initial stiffness).

Static analysis with load control:

eg32gdxl.inp	GK3D2 elements, elastic with damage.
eg32glxl.inp	GK3D2 elements, elastic-plastic (low initial stiffness).
eg38gdxl.inp	GK3D8 elements, elastic with damage.
eg38ghxl.inp	GK3D8 elements, elastic-plastic (high initial stiffness).
eg38gdxl_ctd.inp	GK3D8T elements, elastic with damage.
eg38ghxl_ctd.inp	GK3D8T elements, elastic-plastic (high initial stiffness).
eg3rgdxl.inp	GK3D18 elements, elastic with damage.
eg3rghxl.inp	GK3D18 elements, elastic-plastic (high initial stiffness).
ega4gdxl.inp	GKAX4 elements, elastic with damage.
ega4glxl.inp	GKAX4 elements, elastic-plastic (low initial stiffness).
ege4gdxl.inp	GKPE4 elements, elastic with damage.
ege4ghxl.inp	GKPE4 elements, elastic-plastic (high initial stiffness).
egl6gdxl.inp	GK3D6L elements, elastic with damage.
egl6ghxl.inp	GK3D6L elements, elastic-plastic (high initial stiffness).
egs4gdxl.inp	GKPS4 elements, elastic with damage.
egs4ghxl.inp	GKPS4 elements, elastic-plastic (high initial stiffness).

Static analysis with displacement control:

eg22gdxd.inp	GK2D2 elements, elastic with damage.
eg22glxd.inp	GK2D2 elements, elastic-plastic (low initial stiffness).
eg36gdxd.inp	GK3D6 elements, elastic with damage.
eg36glxd.inp	GK3D6 elements, elastic-plastic (low initial stiffness).
eg36gdxd_ctd.inp	GK3D6T elements, elastic with damage.
eg36glxd_ctd.inp	GK3D6T elements, elastic-plastic (low initial stiffness).
eg3agdxd.inp	GK3D12M elements, elastic with damage.
eg3aglxd.inp	GK3D12M elements, elastic-plastic (low initial stiffness).
ega6gdxd.inp	GKAX6 elements, elastic with damage.
ega6glxd.inp	GKAX6 elements, elastic-plastic (low initial stiffness).
ege6gdxd.inp	GKPE6 elements, elastic with damage.
ege6ghxd.inp	GKPE6 elements, elastic-plastic (high initial stiffness).
egl4gdxd.inp	GK3D4L elements, elastic with damage.
egl4glxd.inp	GK3D4L elements, elastic-plastic (low initial stiffness).
egs6gdxd.inp	GKPS6 elements, elastic with damage.
egs6glxd.inp	GKPS6 elements, elastic-plastic (low initial stiffness).

Static analysis with load control enforced through a contact pair:

eg24ndxk.inp	GKPS4N elements, elastic with damage.
eg24nhxk.inp	GKPS4N elements, elastic-plastic (high initial stiffness).
eg32ndxk.inp	GK3D2N elements, elastic with damage.
eg32nhxk.inp	GK3D2N elements, elastic-plastic (high initial stiffness).

eg38ndxk.inp	GK3D8N elements, elastic with damage.
eg38nhxk.inp	GK3D8N elements, elastic-plastic (high initial stiffness).
eg38nhxk_auglagr.inp	GK3D8N elements, elastic-plastic (high initial stiffness), augmented Lagrangian contact model.
eg38ndxk_ctd.inp	GK3D8TN elements, elastic with damage.
eg38nhxk_ctd.inp	GK3D8TN elements, elastic-plastic (high initial stiffness).
eg38nhxk_aug_ctd.inp	GK3D8TN elements, elastic-plastic (high initial stiffness), augmented Lagrangian contact model.
eg3rndxk.inp	GK3D18N elements, elastic with damage.
eg3rnhxk.inp	GK3D18N elements, elastic-plastic (high initial stiffness).
ega4ndxk.inp	GKAX4N elements, elastic with damage.
ega4nhxk.inp	GKAX4N elements, elastic-plastic (high initial stiffness).
egl6ndxk.inp	GK3D6LN elements, elastic with damage.
egl6nhxk.inp	GK3D6LN elements, elastic-plastic (high initial stiffness).

Static analysis with displacement control enforced through a contact pair:

eg22ndxc.inp	GK2D2N elements, elastic with damage.
eg22nhxc.inp	GK2D2N elements, elastic-plastic (high initial stiffness).
eg26ndxc.inp	GKPS6N elements, elastic with damage.
eg26nhxc.inp	GKPS6N elements, elastic-plastic (high initial stiffness).
eg36ndxc.inp	GK3D6N elements, elastic with damage.
eg36nhxc.inp	GK3D6N elements, elastic-plastic (high initial stiffness).
eg36ndxc_ctd.inp	GK3D6TN elements, elastic with damage.
eg36nhxc_ctd.inp	GK3D6TN elements, elastic-plastic (high initial stiffness).
eg3andxc.inp	GK3D12MN elements, elastic with damage.
eg3andxc_auglagr.inp	GK3D12MN elements, elastic with damage, augmented Lagrangian contact model.
eg3anlxc.inp	GK3D12MN elements, elastic-plastic (low initial stiffness).
eg3anlxc_auglagr.inp	GK3D12MN elements, elastic-plastic (low initial stiffness), augmented Lagrangian contact model.
ega6ndxc.inp	GKAX6N elements, elastic with damage.
ega6nhxc.inp	GKAX6N elements, elastic-plastic (high initial stiffness).
egl4ndxc.inp	GK3D4LN elements, elastic with damage.
egl4nhxc.inp	GK3D4LN elements, elastic-plastic (high initial stiffness).

Steady-state dynamic analysis with displacement control about different preloaded base states:

gasket2d_visc1_str.inp	Two-dimensional gasket elements, elastic with damage and viscoelastic (defined in terms of storage and loss moduli ratios).
gasket2d_visc2_str.inp	Two-dimensional gasket elements, elastic with damage and viscoelastic (defined directly in terms of storage and loss moduli).
gasket3d_visc1_str.inp	Three-dimensional gasket elements, elastic with damage and viscoelastic (defined in terms of storage and loss moduli ratios).
gasket3d_visc2_str.inp	Three-dimensional gasket elements, elastic with damage and viscoelastic (defined directly in terms of storage and loss moduli).
gasketaxi_visc1_str.inp	Axisymmetric gasket elements, elastic with damage and viscoelastic (defined in terms of storage and loss moduli ratios).

gasketaxi_visc2_str.inp	Axisymmetric gasket elements, elastic with damage and viscoelastic (defined directly in terms of storage and loss moduli).
gasketline3d_visc1_force.inp	Three-dimensional line gasket elements, elastic with damage and viscoelastic (defined in terms of storage and loss moduli ratios).
gasketline3d_visc2_force.inp	Three-dimensional line gasket elements, elastic with damage and viscoelastic (defined directly in terms of storage and loss moduli).
gasketlink2d_visc1_force.inp	Two-dimensional link gasket elements, elastic with damage and viscoelastic (defined in terms of storage and loss moduli ratios).
gasketlink2d_visc2_force.inp	Two-dimensional link gasket elements, elastic with damage and viscoelastic (defined directly in terms of storage and loss moduli).
gasketlink3d_visc1_force.inp	Three-dimensional link gasket elements, elastic with damage and viscoelastic (defined in terms of storage and loss moduli ratios).
gasketlink3d_visc2_force.inp	Three-dimensional link gasket elements, elastic with damage and viscoelastic (defined directly in terms of storage and loss moduli).
gasketlinkaxi_visc1_force.inp	Axisymmetric link gasket elements, elastic with damage and viscoelastic (defined in terms of storage and loss moduli ratios).
gasketlinkaxi_visc2_force.inp	Axisymmetric link gasket elements, elastic with damage and viscoelastic (defined directly in terms of storage and loss moduli).

Field expansion test:

fieldexp-gasket.inp	Fully constrained GK2D2 element including field expansion in the thickness-direction behavior. The field expansion is driven by two different field variables. Both general static and linear perturbation steps are tested.

Gasket element assembly

Products: Abaqus/Standard

Overview

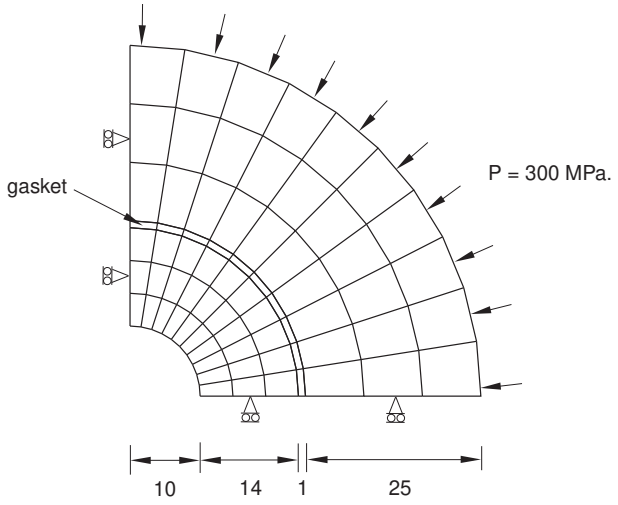
This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

GK3D6 GK3D6N GK3D6T GK3D6TN GK3D8 GK3D8N GK3D8T GK3D8TN GK3D12M GK3D12MN GK3D18 GK3D18N
GKPE4 GKPE6 GKPS4 GKPS4N GKPS6 GKPS6N

Problem description

The different methods for joining gaskets to the remainder of the mesh are tested in this section. A 1 mm thick cylindrical gasket, sandwiched between two coaxial cylindrical tubes, is considered. The inner cylindrical tube has an inner radius of 10 mm and an outer radius of 24 mm, whereas the outer cylindrical tube has an inner radius of 25 mm and an outer radius of 50 mm. The outer cylinder is subjected to a pressure of 300 MPa on the outer surface.



The problem is modeled either as a plane strain problem, a plane stress problem, or a three-dimensional problem. Using symmetry conditions, a quarter of the geometry is modeled. A unit-thickness slice is modeled in all cases. The thickness direction of all gasket elements is the positive radial direction. Therefore, for any gasket element the surface closest to the cylindrical axis represents the bottom surface and the farthest surface represents the top surface. The thickness direction is specified with the user-specified normal definition at the symmetry planes. The gasket is modeled either as a single- or two-layer gasket. The gasket is joined to the rest of the model by using shared nodes, TIE and PINMPCs, or contact pairs with no friction. When contact pairs are used, the input files demonstrate the use of general and tied contact conditions.

Different element types are used to model the tubes and the gasket, and suitable methods are chosen to join the two materials. For example, element types C3D27R and GK3D18 are used with shared nodes, whereas C3D20R and GK3D18N are used with contact pairs. A user-defined coordinate system is used to specify the local 2 and

3 directions for all three-dimensional gasket elements. No mesh convergence studies of the solution have been performed.

Material: Cylindrical tubes: Young's modulus = 2.0×10^5 MPa, Poisson's ratio = 0.3.

Gasket: The thickness-direction behavior is linear elastic such that, for a gasket of unit thickness, the pressure is 400 MPa for a closure of 0.002 mm. The damage model with no unloading curve is used to specify this behavior. The membrane behavior of the gasket has the same Young's modulus and Poisson's ratio as the cylindrical tubes. Depending on the type of gasket elements used and the method used to join them to the cylindrical tubes, the membrane behavior may or may not be used. For the coupled thermal-stress models, the gap conductance is defined as a function of closure for gasket elements: 1.123 for zero closure and 1.522 for 1.0 closure. For coupled solid elements, density=2000, conductivity=1., and specific heat=0.001.

Results and discussion

The generalized strains in the gasket elements are consistent with the displacements of their top and bottom surfaces, and the generalized stresses are obtained correctly from the generalized strains through the specified gasket behavior.

Input files

ege4gdxf.inp	GKPE4 elements.
ege6gdxf.inp	GKPE6 elements.
egs4gdxf.inp	GKPS4 elements.
egs4gdxs.inp	GKPS4 elements; SOLID ELEMENT NUMBERING.
eg24ndxf.inp	GKPS4N elements.
egs6gdxf.inp	GKPS6 elements.
egs6gdxs.inp	GKPS6 elements; SOLID ELEMENT NUMBERING.
eg26ndxf.inp	GKPS6N elements.
eg36gdxf.inp	GK3D6 elements.
eg36ndxf.inp	GK3D6N elements.
eg38gdxf.inp	GK3D8 elements.
eg38ndxf.inp	GK3D8N elements.
eg36gdxf_ctd.inp	GK3D6T elements.
eg36ndxf_ctd.inp	GK3D6TN elements.
eg38gdxf_ctd.inp	GK3D8T elements.
eg38ndxf_ctd.inp	GK3D8TN elements.
eg3agdxf.inp	GK3D12M elements.
eg3agdxs.inp	GK3D12M elements; SOLID ELEMENT NUMBERING.
eg3andxf.inp	GK3D12MN elements.
eg3rgdxf.inp	GK3D18 elements.
eg3rgdxm.inp	GK3D18 elements; generation of missing nodes.
eg3rgdxs.inp	GK3D18 elements; SOLID ELEMENT NUMBERING.
eg3rndxf.inp	GK3D18N elements.
eg3rndxm.inp	GK3D18N elements; generation of missing nodes.

Cohesive elements

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Features tested

This section provides verification for the following:

- Element properties for cohesive elements.
- Material and contact properties to define damage initiation.
- Material properties to define damage evolution.
- Use of cohesive elements in symmetric modeling.

The pressure continuity is also verified for the undamaged pore pressure cohesive elements in Abaqus/Standard.

Element kinematics

Elements tested

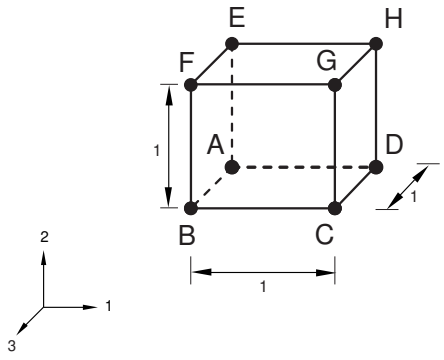
COH3D8 COH3D6 COH2D4 COHAX4

Problem description

The following three types of constitutive response for cohesive elements are verified in this test:

- Cohesive elements used as gaskets or small adhesive patches.
- A finite-thickness adhesive layer modeled using a continuum-based constitutive response.
- A negligibly thin layer of adhesive modeled using a traction-separation-based response.

Each response is verified for deformation in pure normal and two pure shear modes (one shear mode for two-dimensional and axisymmetric elements) by applying appropriate displacement boundary conditions.



Model: This test comprises single-element models, the geometry of which is defined so that the initial thickness is 1.0 for each case. The thickness direction for the elements is set to the global 1-direction, except for COH3D6, for which the thickness direction is set to the default direction.

Material: The response of cohesive elements is tested for the following material models:

- Linear elastic
- Hyperelastic
- Hyperfoam
- Mises plasticity
- Drucker-Prager plasticity

Boundary conditions:

Pure normal mode:

$$u_1^C = u_1^D = u_1^G = u_1^H = 1.0 u_1^A = u_1^B = u_1^E = u_1^F = -1.0$$

Pure shear in the first shear direction:

$$u_3^C = u_3^D = u_3^G = u_3^H = 1.0 u_3^A = u_3^B = u_3^E = u_3^F = -1.0$$

Pure shear in the second shear direction:

$$u_2^C = u_2^D = u_2^G = u_2^H = 1.0 u_2^A = u_2^B = u_2^E = u_2^F = -1.0$$

All degrees of freedom other than those listed above are fixed.

Results and discussion

The response of the cohesive elements matches the analytical results.

Input files

Abaqus/Standard input files

lk_coh3d8_ts_stack1_std.inp	TRACTION SEPARATION response for COH3D8.
lk_coh3d8_co_stack1_std.inp	CONTINUUM response for COH3D8.
lk_coh3d8_gk_stack1_std.inp	GASKET response for COH3D8.
lk_coh3d6_ts_std.inp	TRACTION SEPARATION response for COH3D6.
lk_coh3d6_co_std.inp	CONTINUUM response for COH3D6.
lk_coh3d6_gk_std.inp	GASKET response for COH3D6.
lk_coh2d4_ts_stack1_std.inp	TRACTION SEPARATION response for COH2D4.
lk_coh2d4_co_stack1_std.inp	CONTINUUM response for COH2D4.
lk_coh2d4_gk_stack1_std.inp	GASKET response for COH2D4.
lk_cohax4_ts_stack1_std.inp	TRACTION SEPARATION response for COHAX4.
lk_cohax4_co_stack1_std.inp	CONTINUUM response for COHAX4.

lk_cohax4_gk_stack1_std.inp	GASKET response for COHAX4.
coh_co_hyper_std.inp	CONTINUUM response for COH3D8, COH3D6, COH2D4, COHAX4 with hyperelasticity.
coh_gk_hyper_std.inp	GASKET response for COH3D8, COH3D6, COH2D4, COHAX4 with hyperelasticity.
coh_co_hyperfoam_std.inp	CONTINUUM response for COH3D8, COH3D6, COH2D4, COHAX4 with hyperfoam material.
coh_gk_hyperfoam_std.inp	GASKET response for COH3D8, COH3D6, COH2D4, COHAX4 with hyperfoam material.
coh_co_mises_std.inp	CONTINUUM response for COH3D8, COH3D6, COH2D4, COHAX4 with Mises plasticity.
coh_gk_mises_std.inp	GASKET response for COH3D8, COH3D6, COH2D4, COHAX4 with Mises plasticity.
coh_co_dp_std.inp	CONTINUUM response for COH3D8, COH3D6, COH2D4, COHAX4 with Drucker-Prager plasticity.
coh_transshear_std.inp	COH3D8, COH3D6, COH2D4, COHAX4 with uncoupled transverse shear stiffness specified using the **TRANSVERSE SHEAR STIFFNESS* option.

Abaqus/Explicit input files

lk_coh3d8_ts_stack1_xpl.inp	TRACTION SEPARATION response for COH3D8.
lk_coh3d8_co_stack1_xpl.inp	CONTINUUM response for COH3D8.
lk_coh3d8_gk_stack1_xpl.inp	GASKET response for COH3D8.
lk_coh3d6_ts_xpl.inp	TRACTION SEPARATION response for COH3D6.
lk_coh3d6_co_xpl.inp	CONTINUUM response for COH3D6.
lk_coh3d6_gk_xpl.inp	GASKET response for COH3D6.
lk_coh2d4_ts_stack1_xpl.inp	TRACTION SEPARATION response for COH2D4.
lk_coh2d4_co_stack1_xpl.inp	CONTINUUM response for COH2D4.
lk_coh2d4_gk_stack1_xpl.inp	GASKET response for COH2D4.
lk_cohax4_ts_stack1_xpl.inp	TRACTION SEPARATION response for COHAX4.
lk_cohax4_co_stack1_xpl.inp	CONTINUUM response for COHAX4.
lk_cohax4_gk_stack1_xpl.inp	GASKET response for COHAX4.
coh_co_hyper_xpl.inp	CONTINUUM response for COH3D8, COH3D6, COH2D4, COHAX4 with hyperelasticity.
coh_gk_hyper_xpl.inp	GASKET response for COH3D8, COH3D6, COH2D4, COHAX4 with hyperelasticity.
coh_co_hyperfoam_xpl.inp	CONTINUUM response for COH3D8, COH3D6, COH2D4, COHAX4 with hyperfoam material.
coh_gk_hyperfoam_xpl.inp	GASKET response for COH3D8, COH3D6, COH2D4, COHAX4 with hyperfoam material.
coh_transshear_xpl.inp	COH3D8, COH3D6, COH2D4, COHAX4 with uncoupled transverse shear stiffness specified using the **TRANSVERSE SHEAR STIFFNESS* option.

Damage modeling verification

Elements tested

COH3D8 COH3D6 COH2D4 COHAX4 COH3D8P COH3D6P COH2D4P COHAX4P

Problem description

This test verifies damage modeling for cohesive elements using different damage initiation criteria and damage evolution laws to simulate the failure of cohesive layers. A linear elastic material model is used to verify the damage initiation criteria based on the maximum nominal strain for cohesive elements and based on the quadratic traction-interaction for cohesive elements. Damage initiation criteria based on the ductile strain and based on the shear failure strain are tested with Mises and Drucker-Prager plasticity, respectively.

Damage evolution is defined based on either effective displacement or energy dissipated. Linear, exponential, and tabular softening laws are defined to specify the nature of the evolution of the damage variable. Each damage model is verified for damage in pure normal and two pure shear modes (one shear mode for two-dimensional and axisymmetric elements). The dependence of damage evolution on the mode mix measure specified in tabular, power law, or Benzeggagh-Kenane form is also considered in this test. In addition, the test verifies the overall damage of cohesive elements when multiple damage initiation criteria are active for the same material definition.

Results and discussion

Degradation of the response of a cohesive element begins when the specified damage initiation criterion is met. The damage variable evolves according to the evolution law specified in terms of displacement or energy dissipation.

Input files

Abaqus/Standard input files

coh3d8_mxe_damdisp_softlin_std.inp	MAXE damage initiation, displacement-based damage evolution with LINEAR softening for COH3D8.
coh3d8_qds_damdisp_softlin_std.inp	QUADS damage initiation, displacement-based damage evolution with LINEAR softening for COH3D8.
coh3d8_mxe_damdisp_softexp_std.inp	MAXE damage initiation, displacement-based damage evolution with EXPONENTIAL softening for COH3D8.
coh3d8_qds_damdisp_softexp_std.inp	QUADS damage initiation, displacement-based damage evolution with EXPONENTIAL softening for COH3D8.
coh3d8_mxe_damdisp_softtab_std.inp	MAXE damage initiation, displacement-based damage evolution with TABULAR softening for COH3D8.
coh3d8_qds_damdisp_softtab_std.inp	QUADS damage initiation, displacement-based damage evolution with TABULAR softening for COH3D8.
coh3d8_mxe_damener_softlin_std.inp	MAXE damage initiation, energy-based damage evolution with LINEAR softening for COH3D8.
coh3d8_qds_damener_softlin_std.inp	QUADS damage initiation, energy-based damage evolution with LINEAR softening for COH3D8.
coh3d8_mxe_damener_softexp_std.inp	MAXE damage initiation, energy-based damage evolution with EXPONENTIAL softening for COH3D8.
coh3d8_qds_damener_softexp_std.inp	QUADS damage initiation, energy-based damage evolution with EXPONENTIAL softening for COH3D8.
coh3d8_nomodemix_std.inp	Damage evolution independent of mode mix for COH3D8.

coh3d8p_mxe_damdisp_softlin_std.inp	MAXE damage initiation, displacement-based damage evolution with LINEAR softening for COH3D8P.
coh3d6_mxe_damdisp_softlin_std.inp	MAXE damage initiation, displacement-based damage evolution with LINEAR softening for COH3D6.
coh3d6_qds_damdisp_softlin_std.inp	QUADS damage initiation, displacement-based damage evolution with LINEAR softening for COH3D6.
coh3d6_mxe_damdisp_softexp_std.inp	MAXE damage initiation, displacement-based damage evolution with EXPONENTIAL softening for COH3D6.
coh3d6_qds_damdisp_softexp_std.inp	QUADS damage initiation, displacement-based damage evolution with EXPONENTIAL softening for COH3D6.
coh3d6_mxe_damdisp_softtab_std.inp	MAXE damage initiation, displacement-based damage evolution with TABULAR softening for COH3D6.
coh3d6_qds_damdisp_softtab_std.inp	QUADS damage initiation, displacement-based damage evolution with TABULAR softening for COH3D6.
coh3d6_mxe_damener_softlin_std.inp	MAXE damage initiation, energy-based damage evolution with LINEAR softening for COH3D6.
coh3d6_qds_damener_softlin_std.inp	QUADS damage initiation, energy-based damage evolution with LINEAR softening for COH3D6.
coh3d6_mxe_damener_softexp_std.inp	MAXE damage initiation, energy-based damage evolution with EXPONENTIAL softening for COH3D6.
coh3d6_qds_damener_softexp_std.inp	QUADS damage initiation, energy-based damage evolution with EXPONENTIAL softening for COH3D6.
coh3d6_nomodemix_std.inp	Damage evolution independent of mode mix for COH3D6.
coh3d6p_mxe_damdisp_softlin_std.inp	MAXE damage initiation, displacement-based damage evolution with LINEAR softening for COH3D6P.
coh2d4_mxe_damdisp_softlin_std.inp	MAXE damage initiation, displacement-based damage evolution with LINEAR softening for COH2D4.
coh2d4_qds_damdisp_softlin_std.inp	QUADS damage initiation, displacement-based damage evolution with LINEAR softening for COH2D4.
coh2d4_mxe_damdisp_softexp_std.inp	MAXE damage initiation, displacement-based damage evolution with EXPONENTIAL softening for COH2D4.
coh2d4_qds_damdisp_softexp_std.inp	QUADS damage initiation, displacement-based damage evolution with EXPONENTIAL softening for COH2D4.
coh2d4_mxe_damdisp_softtab_std.inp	MAXE damage initiation, displacement-based damage evolution with TABULAR softening for COH2D4.
coh2d4_qds_damdisp_softtab_std.inp	QUADS damage initiation, displacement-based damage evolution with TABULAR softening for COH2D4.
coh2d4_mxe_damener_softlin_std.inp	MAXE damage initiation, energy-based damage evolution with LINEAR softening for COH2D4.
coh2d4_qds_damener_softlin_std.inp	QUADS damage initiation, energy-based damage evolution with LINEAR softening for COH2D4.
coh2d4_mxe_damener_softexp_std.inp	MAXE damage initiation, energy-based damage evolution with EXPONENTIAL softening for COH2D4.
coh2d4_qds_damener_softexp_std.inp	QUADS damage initiation, energy-based damage evolution with EXPONENTIAL softening for COH2D4.
coh2d4_nomodemix_std.inp	Damage evolution independent of mode mix for COH2D4.
coh2d4p_mxe_damdisp_softlin_std.inp	MAXE damage initiation, displacement-based damage evolution with LINEAR softening for COH2D4P.

cohax4_mxe_damdisp_softlin_std.inp	MAXE damage initiation, displacement-based damage evolution with LINEAR softening for COHAX4.
cohax4_qds_damdisp_softlin_std.inp	QUADS damage initiation, displacement-based damage evolution with LINEAR softening for COHAX4.
cohax4_mxe_damdisp_softexp_std.inp	MAXE damage initiation, displacement-based damage evolution with EXPONENTIAL softening for COHAX4.
cohax4_qds_damdisp_softexp_std.inp	QUADS damage initiation, displacement-based damage evolution with EXPONENTIAL softening for COHAX4.
cohax4_mxe_damdisp_softtab_std.inp	MAXE damage initiation, displacement-based damage evolution with TABULAR softening for COHAX4.
cohax4_qds_damdisp_softtab_std.inp	QUADS damage initiation, displacement-based damage evolution with TABULAR softening for COHAX4.
cohax4_mxe_damener_softlin_std.inp	MAXE damage initiation, energy-based damage evolution with LINEAR softening for COHAX4.
cohax4_qds_damener_softlin_std.inp	QUADS damage initiation, energy-based damage evolution with LINEAR softening for COHAX4.
cohax4_mxe_damener_softexp_std.inp	MAXE damage initiation, energy-based damage evolution with EXPONENTIAL softening for COHAX4.
cohax4_qds_damener_softexp_std.inp	QUADS damage initiation, energy-based damage evolution with EXPONENTIAL softening for COHAX4.
cohax4_nomodemix_std.inp	Damage evolution independent of mode mix for COHAX4.
cohax4p_mxe_damdisp_softlin_std.inp	MAXE damage initiation, displacement-based damage evolution with LINEAR softening for COHAX4P.
coh3d8_coupled_multi_std.inp	COH3D8 with multiple damage models and coupled traction-separation behavior.
coh2d4_coupled_multi_std.inp	COH2D4 with multiple damage models and coupled traction-separation behavior.
coh3d8_ts_dam_loadcycle_std.inp	COH3D8 subjected to loading and unloading in pure normal (both tension and compression) and pure shear modes after partial damage.
coh2d4_damdisp_mixtrac_std.inp	Displacement-based damage evolution with traction-dependent mode mix measure for COH2D4.
coh2d4_damdisp_mixener_std.inp	Displacement-based damage evolution with energy-dependent mode mix measure for COH2D4.
coh2d4_damener_mixtrac_std.inp	Energy-based damage evolution with traction-dependent mode mix measure for COH2D4.
coh2d4_damener_mixener_std.inp	Energy-based damage evolution with energy-dependent mode mix measure for COH2D4.
coh3d8_damdisp_mixtrac_std.inp	Displacement-based damage evolution with traction-dependent mode mix measure for COH3D8.
coh3d8_damdisp_mixener_std.inp	Displacement-based damage evolution with energy-dependent mode mix measure for COH3D8.
coh3d8_damener_mixtrac_std.inp	Energy-based damage evolution with traction-dependent mode mix measure for COH3D8.
coh3d8_damener_mixener_std.inp	Energy-based damage evolution with energy-dependent mode mix measure for COH3D8.
coh_co_misesduct_std.inp	DUCTILE damage initiation; CONTINUUM response for COH3D8, COH3D6, COH2D4, COHAX4 with Mises plasticity.

coh_co_misesduct_hc_std.inp	HC damage initiation; CONTINUUM response for COH3D8, COH3D6, COH2D4, COHAX4 with Mises plasticity.
coh_co_misesshear_std.inp	SHEAR damage initiation; CONTINUUM response for COH3D8, COH3D6, COH2D4, COHAX4 with Mises plasticity.
coh_co_dpduct_std.inp	DUCTILE damage initiation; CONTINUUM response for COH3D8, COH3D6, COH2D4, COHAX4 with Drucker-Prager plasticity.
coh_co_dpduct_hc_std.inp	HC damage initiation; CONTINUUM response for COH3D8, COH3D6, COH2D4, COHAX4 with Drucker-Prager plasticity.
coh_co_dpshear_std.inp	SHEAR damage initiation; CONTINUUM response for COH3D8, COH3D6, COH2D4, COHAX4 with Drucker-Prager plasticity.

Abaqus/Explicit input files

coh3d8_mxe_damdisp_softlin_xpl.inp	MAXE damage initiation, displacement-based damage evolution with LINEAR softening for COH3D8.
coh3d8_qds_damdisp_softlin_xpl.inp	QUADS damage initiation, displacement-based damage evolution with LINEAR softening for COH3D8.
coh3d8_mxe_damdisp_softexp_xpl.inp	MAXE damage initiation, displacement-based damage evolution with EXPONENTIAL softening for COH3D8.
coh3d8_qds_damdisp_softexp_xpl.inp	QUADS damage initiation, displacement-based damage evolution with EXPONENTIAL softening for COH3D8.
coh3d8_mxe_damdisp_softtab_xpl.inp	MAXE damage initiation, displacement-based damage evolution with TABULAR softening for COH3D8.
coh3d8_qds_damdisp_softtab_xpl.inp	QUADS damage initiation, displacement-based damage evolution with TABULAR softening for COH3D8.
coh3d8_mxe_damener_softlin_xpl.inp	MAXE damage initiation, energy-based damage evolution with LINEAR softening for COH3D8.
coh3d8_qds_damener_softlin_xpl.inp	QUADS damage initiation, energy-based damage evolution with LINEAR softening for COH3D8.
coh3d8_mxe_damener_softexp_xpl.inp	MAXE damage initiation, energy-based damage evolution with EXPONENTIAL softening for COH3D8.
coh3d8_qds_damener_softexp_xpl.inp	QUADS damage initiation, energy-based damage evolution with EXPONENTIAL softening for COH3D8.
coh3d8_nomodemix_xpl.inp	Damage evolution independent of mode mix for COH3D8.
coh3d6_mxe_damdisp_softlin_xpl.inp	MAXE damage initiation, displacement-based damage evolution with LINEAR softening for COH3D6.
coh3d6_qds_damdisp_softlin_xpl.inp	QUADS damage initiation, displacement-based damage evolution with LINEAR softening for COH3D6.
coh3d6_mxe_damdisp_softexp_xpl.inp	MAXE damage initiation, displacement-based damage evolution with EXPONENTIAL softening for COH3D6.
coh3d6_qds_damdisp_softexp_xpl.inp	QUADS damage initiation, displacement-based damage evolution with EXPONENTIAL softening for COH3D6.
coh3d6_mxe_damdisp_softtab_xpl.inp	MAXE damage initiation, displacement-based damage evolution with TABULAR softening for COH3D6.
coh3d6_qds_damdisp_softtab_xpl.inp	QUADS damage initiation, displacement-based damage evolution with TABULAR softening for COH3D6.
coh3d6_mxe_damener_softlin_xpl.inp	MAXE damage initiation, energy-based damage evolution with LINEAR softening for COH3D6.

coh3d6_qds_damener_softlin_xpl.inp	QUADS damage initiation, energy-based damage evolution with LINEAR softening for COH3D6.
coh3d6_mxe_damener_softexp_xpl.inp	MAXE damage initiation, energy-based damage evolution with EXPONENTIAL softening for COH3D6.
coh3d6_qds_damener_softexp_xpl.inp	QUADS damage initiation, energy-based damage evolution with EXPONENTIAL softening for COH3D6.
coh3d6_nomodemix_xpl.inp	Damage evolution independent of mode mix for COH3D6.
coh2d4_mxe_damdisp_softlin_xpl.inp	MAXE damage initiation, displacement-based damage evolution with LINEAR softening for COH2D4.
coh2d4_qds_damdisp_softlin_xpl.inp	QUADS damage initiation, displacement-based damage evolution with LINEAR softening for COH2D4.
coh2d4_mxe_damdisp_softexp_xpl.inp	MAXE damage initiation, displacement-based damage evolution with EXPONENTIAL softening for COH2D4.
coh2d4_qds_damdisp_softexp_xpl.inp	QUADS damage initiation, displacement-based damage evolution with EXPONENTIAL softening for COH2D4.
coh2d4_mxe_damdisp_softtab_xpl.inp	MAXE damage initiation, displacement-based damage evolution with TABULAR softening for COH2D4.
coh2d4_qds_damdisp_softtab_xpl.inp	QUADS damage initiation, displacement-based damage evolution with TABULAR softening for COH2D4.
coh2d4_mxe_damener_softlin_xpl.inp	MAXE damage initiation, energy-based damage evolution with LINEAR softening for COH2D4.
coh2d4_qds_damener_softlin_xpl.inp	QUADS damage initiation, energy-based damage evolution with LINEAR-softening for COH2D4.
coh2d4_mxe_damener_softexp_xpl.inp	MAXE damage initiation, energy-based damage evolution with EXPONENTIAL softening for COH2D4.
coh2d4_qds_damener_softexp_xpl.inp	QUADS damage initiation, energy-based damage evolution with EXPONENTIAL softening for COH2D4.
coh2d4_nomodemix_xpl.inp	Damage evolution independent of mode mix for COH2D4.
cohax4_mxe_damdisp_softlin_xpl.inp	MAXE damage initiation, displacement-based damage evolution with LINEAR softening for COHAX4.
cohax4_qds_damdisp_softlin_xpl.inp	QUADS damage initiation, displacement-based damage evolution with LINEAR softening for COHAX4.
cohax4_mxe_damdisp_softexp_xpl.inp	MAXE damage initiation, displacement-based damage evolution with EXPONENTIAL softening for COHAX4.
cohax4_qds_damdisp_softexp_xpl.inp	QUADS damage initiation, displacement-based damage evolution with EXPONENTIAL softening for COHAX4.
cohax4_mxe_damdisp_softtab_xpl.inp	MAXE damage initiation, displacement-based damage evolution with TABULAR softening for COHAX4.
cohax4_qds_damdisp_softtab_xpl.inp	QUADS damage initiation, displacement-based damage evolution with TABULAR softening for COHAX4.
cohax4_mxe_damener_softlin_xpl.inp	MAXE damage initiation, energy-based damage evolution with LINEAR softening for COHAX4.
cohax4_qds_damener_softlin_xpl.inp	QUADS damage initiation, energy-based damage evolution with LINEAR softening for COHAX4.
cohax4_mxe_damener_softexp_xpl.inp	MAXE damage initiation, energy-based damage evolution with EXPONENTIAL softening for COHAX4.
cohax4_qds_damener_softexp_xpl.inp	QUADS damage initiation, energy-based damage evolution with EXPONENTIAL softening for COHAX4.

cohax4_nomodemix_xpl.inp	Damage evolution independent of mode mix for COHAX4.
coh_co_misesduct_xpl.inp	DUCTILE damage initiation; CONTINUUM response for COH3D8, COH3D6, COH2D4, COHAX4 with Mises plasticity.
coh_gk_misesduct_xpl.inp	DUCTILE damage initiation; GASKET response for COH3D8, COH3D6, COH2D4, COHAX4 with Mises plasticity.
coh_co_misesshear_xpl.inp	SHEAR damage initiation; CONTINUUM response for COH3D8, COH3D6, COH2D4, COHAX4 with Mises plasticity.
coh_gk_misesshear_xpl.inp	SHEAR damage initiation; GASKET response for COH3D8, COH3D6, COH2D4, COHAX4 with Mises plasticity.
coh_co_dpduct_xpl.inp	DUCTILE damage initiation; CONTINUUM response for COH3D8, COH3D6, COH2D4, COHAX4 with Drucker-Prager plasticity.
coh_co_dpshear_xpl.inp	SHEAR damage initiation; CONTINUUM response for COH3D8, COH3D6, COH2D4, COHAX4 with Drucker-Prager plasticity.
coh3d8_coupled_multi_xpl.inp	COH3D8 with multiple damage models and coupled traction-separation behavior.
coh2d4_coupled_multi_xpl.inp	COH2D4 with multiple damage models and coupled traction-separation behavior.
coh3d8_ts_dam_loadcycle_xpl.inp	COH3D8 subjected to loading and unloading in pure normal (both tension and compression) and pure shear modes after partial damage.
coh2d4_damdisp_mixtrac_xpl.inp	Displacement-based damage evolution with traction-dependent mode mix measure for COH2D4.
coh2d4_damdisp_mixener_xpl.inp	Displacement-based damage evolution with energy-dependent mode mix measure for COH2D4.
coh2d4_damener_mixtrac_xpl.inp	Energy-based damage evolution with traction-dependent mode mix measure for COH2D4.
coh2d4_damener_mixener_xpl.inp	Energy-based damage evolution with energy-dependent mode mix measure for COH2D4.
coh3d8_damdisp_mixtrac_xpl.inp	Displacement-based damage evolution with traction-dependent mode mix measure for COH3D8.
coh3d8_damdisp_mixener_xpl.inp	Displacement-based damage evolution with energy-dependent mode mix measure for COH3D8.
coh3d8_damener_mixtrac_xpl.inp	Energy-based damage evolution with traction-dependent mode mix measure for COH3D8.
coh3d8_damener_mixener_xpl.inp	Energy-based damage evolution with energy-dependent mode mix measure for COH3D8.

Pressure continuity for pore pressure cohesive elements

Elements tested

COH3D8P COH3D6P COH2D4P COHAX4P

Problem description

This test verifies the pressure continuity for pore pressure cohesive elements without damage.

The model contains two blocks meshed by using pore pressure solid elements. One block is on the top, while another one is on the bottom. They are connected to each other through a layer of pore pressure cohesive elements. No damage is introduced to the pore pressure cohesive elements in the tests. When different pressure is specified at the top and the bottom sides of model, the driven fluid flows smoothly across the layer of cohesive elements

generating the same pressure gradient everywhere. In some tests resistance is introduced to the flow by building a “filter cake” by defining leak-off coefficients for pore pressure cohesive elements. In some tests the solid and cohesive elements have different mesh densities; therefore, a surface-based tie constraint will be used to connect them to each other.

Results and discussion

The smooth variation of pore pressure can be observed crossing the layer of cohesive elements.

Input files

Available only in Abaqus/Standard

coh2d4p_cont.inp	CPE4P with COH2D4P.
coh3d6p_cont.inp	C3D8P with COH3D6P.
coh3d8p_cont.inp	C3D8P with COH3D8P.
cohax4p_cont.inp	CAX4P with COHAX4P.
coh2d4p_cont_leak.inp	CPE4P with COH2D4P using **FLUID LEAKOFF*.
coh3d6p_cont_leak.inp	C3D8P with COH3D6P using **FLUID LEAKOFF*.
coh3d8p_cont_leak.inp	C3D8P with COH3D8P using **FLUID LEAKOFF*.
cohax4p_cont_leak.inp	CAX4P with COHAX4P using **FLUID LEAKOFF*.
coh2d4p_cont_tie.inp	CPE4P with COH2D4P using **TIE*.
coh3d6p_cont_tie.inp	C3D8P with COH3D6P using **TIE*.
coh3d8p_cont_tie.inp	C3D8P with COH3D8P using **TIE*.
cohax4p_cont_tie.inp	CAX4P with COHAX4P using **TIE*.

Using cohesive elements for symmetric modeling

Elements tested

COD3D8P COD3D8PT COH2D4 COH2D4P COH3D8 COH3D8P

Problem description

This test verifies that symmetric models using cohesive elements generate the same results as the corresponding full models.

The full model contains two blocks meshed by using solid elements. One block is on the top, while another one is on the bottom. They are connected to each other through a layer of cohesive elements. In the symmetric models only one half of the structure is used.

Results and discussion

The results using symmetric models agree well with the results of the full models.

Input files

Available only in Abaqus/Standard

rect_coh2d4_full.inp	Full model using CPE4R with COH2D4.
rect_coh2d4_symm.inp	Symmetric model using CPE4R with COH2D4.
rect_coh2d4_half.inp	Half model using CPE4R with COH2D4.

rect_coh3d8_full.inp	Full model using C3D8R with COH3D8.
rect_coh3d8_symm.inp	Symmetric model using C3D8R with COH3D8.
rect_coh3d8_half.inp	Half model using C3D8R with COH3D8.
rect_coh2d4p_full.inp	Full model using CPE4R with COH2D4.
rect_coh2d4p_symm.inp	Symmetric model using CPE4RP with COH2D4P.
rect_coh2d4p_half.inp	Half model using CPE4RP with COH2D4P.
rect_coh3d8p_full.inp	Full model using C3D8RP with COH3D8P.
rect_coh3d8p_symm.inp	Symmetric model using C3D8RP with COH3D8P.
rect_coh3d8p_half.inp	Half model using C3D8RP with COH3D8P.
crack_coh_pipe_full.inp	Full model using cohesive elements with pipe elements.
crack_coh_pipe_symm.inp	Half model using symmetric cohesive elements with pipe elements.
crack_cod3d8pt_fp3d2t_full.inp	Full model using COD3D8PT with FP3D2T elements.
crack_cod3d8pt_fp3d2t_symm.inp	Half model using COD3D8PT with FP3D2T elements.

Miscellaneous tests

Elements tested

COD3D8P COH2D4T

Problem description

Simple tests are created to test heat interaction, heat convection, and different fluids with gap flow.

Results and discussion

In all cases the results are as expected.

Input files

Available only in Abaqus/Standard

coh2d4t_cpe4t.inp	A coupled temperature-displacement model using CPE4T with COH2D4T.
leakoff_cod3d8p_q_p_variable_bingham.inp	A gap flow model using COD3D8P and Bingham Plastic fluid.
leakoff_cod3d8p_q_p_variable_herschel.inp	A gap flow model using COD3D8P and Herschel-Bulkley fluid.

Coriolis loading for direct-solution steady-state dynamic analysis

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

T2D2 T2D3 T3D2 T3D3
CPE3 CPS3 CPE4 CPS4 CPE6 CPS6 CPE6M CPS6M CPE8 CPS8
CPEG3 CPEG4 CPEG6 CPEG8
C3D4 C3D5 C3D6 C3D8 C3D10 C3D10M C3D15 C3D20 C3D27

Problem description

The effect of Coriolis loading in a direct-solution steady-state dynamics analysis is verified. A four-step direct-solution steady-state dynamic procedure is performed on a unit length rod for trusses, on a unit square plate for two-dimensional solids, and on a unit cube for three-dimensional solids. Two elements are used for the triangular and prism element shapes, five elements are used for the tetrahedral element shapes, and one element is used for all other element shapes. The elements are constrained at all nodes and displaced in one degree of freedom: degree of freedom 1 in Steps 1 and 2 and degree of freedom 2 in Steps 3 and 4. Coriolis loading is activated in Steps 2 and 4, and the resulting additional reaction forces and phase shifts are verified by comparing them to analytical values. One representative element type is tested for all solid and truss element classes that can be used in steady-state dynamic analyses and that support Coriolis loading. The use of this feature with submodeling is verified by performing a global and a submodel analysis with CPE4 elements.

Material:

Length for truss models	1
Area	1
Planar dimensions for two-dimensional solids	1×1
Thickness	1
Cubic dimensions for three-dimensional solids	$1 \times 1 \times 1$
Young's modulus	1000.0
Poisson's ratio	0.3
Density	1.0
Damping	α_r=1.0, β_r=0.0
Coriolis loading	1.0
Coriolis axis of rotation	(0, 0, 1) through point (0.5, −10, 0)

Results and discussion

The reaction forces and the phase angle shifts due to the Coriolis loading match the analytical results for all of the elements that are tested.

Input files

ece4sfdg.inp	CPE4 elements, global model.
ece4sfds.inp	CPE4 elements, submodel.
et22sfdc.inp	T2D2 elements.
et23sfdc.inp	T2D3 elements.
et32sfdc.inp	T3D2 elements.
et33sfdc.inp	T3D3 elements.
ece3sfdc.inp	CPE3 elements.
ecs3sfdc.inp	CPS3 elements.
ece4sfdc.inp	CPE4 elements.
ecs4sfdc.inp	CPS4 elements.
ece6sfdc.inp	CPE6 elements.
ecs6sfdc.inp	CPS6 elements.
ece6smdc.inp	CPE6M elements.
ecs6smdc.inp	CPS6M elements.
ece8sfdc.inp	CPE8 elements.
ecs8sfdc.inp	CPS8 elements.
ecg3sfdc.inp	CPEG3 elements.
ecg4sfdc.inp	CPEG4 elements.
ecg6sfdc.inp	CPEG6 elements.
ecg8sfdc.inp	CPEG8 elements.
ec34sfdc.inp	C3D4 elements.
ec35sfdc.inp	C3D5 elements.
ec36sfdc.inp	C3D6 elements.
ec38sfdc.inp	C3D8 elements.
ec3asfdc.inp	C3D10 elements.
ec3asmdc.inp	C3D10M elements.
ec3fsfdc.inp	C3D15 elements.
ec3ksfdc.inp	C3D20 elements.
ec3rsfdc.inp	C3D27 elements.

Pipe-soil interaction elements

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

PSI24 PSI26 PSI34 PSI36

Features tested

The constitutive behavior of the pipe-soil interaction (PSI) elements is tested. The material is defined with different material response in the different directions. The axial and transverse vertical response is symmetric about the origin, while the vertical response uses different behavior for positive and negative relative displacement. An isotropic model, which uses the same material model in all the directions, is also tested.

The local coordinate system procedure is also tested. Temperature and field variable dependence of material properties is tested.

Problem description

The problem consists of a single PSI element subjected to a prescribed displacement history. The far-field edge is fixed, and the displacement history is applied to the pipeline side. The value of the prescribed displacement changes in such a way that the constitutive response corresponding to negative and positive relative displacement is verified.

Each input file contains as many PSI elements as the number of coordinate directions; i.e., two for the two-dimensional elements (PSI24 and PSI26) and three for the three-dimensional elements (PSI34 and PSI36). The prescribed displacement applied to each element is in a different direction. The elements are not connected in any way. Both regular static steps, with small and large displacements, and linear perturbation steps are considered.

Material:

Elastic stiffness in axial direction:	1.0×10^6
Elastic stiffness in vertical direction:	2.0×10^6
Elastic stiffness in horizontal direction:	4.0×10^6

ASCE formulae for sand:

Axial direction:

$\overline{\gamma}$	19000.0
K_0	0.3
δ	30.0°
D	0.6
ε_a	0.003

Vertical direction:

γ	24000.0
$N_v p$	0.5
N_v	0.4
$N_v n$	0.3
$\bar{\varepsilon}_v p$	0.15
$\bar{\varepsilon}_v n$	0.015

Horizontal direction:

N_h	0.25
$\bar{\varepsilon}_h$	0.1

ASCE formulae for clay:

Axial direction:

S	1000
α	1.0
D	0.6
$\bar{\varepsilon}_a$	0.005

Vertical direction:

$N_v p$	0.8
$N_v n$	0.4
$\bar{\varepsilon}_v p$	0.15
$\bar{\varepsilon}_v n$	0.1

Horizontal direction:

N_h	0.25
$\bar{\varepsilon}_h$	0.1

Results and discussion

The forces applied to the pipeline match the analytical values.

Input files

Linear material behavior:

epsi24ls1.inp	PSI24 element with small displacements.
epsi24ls2.inp	PSI24 element with user-defined orientation, small displacements.
epsi24ls3.inp	PSI24 element with user-defined orientation, unsymmetric stiffness, small displacements.

epsi34ls1.inp	PSI34 element with small displacements, isotropic behavior.
epsi24ln3.inp	PSI24 element with field variable dependence, large displacements, isotropic behavior.
epsi26ln1.inp	PSI26 element with large displacements.
epsi26ln2.inp	PSI26 element with user-defined orientation, unsymmetric stiffness, large displacements.
epsi36lp1.inp	PSI36 element with perturbations.
epsi36ln1.inp	PSI36 element with temperature dependence, large displacements.

Nonlinear material behavior:

epsi24ns1.inp	PSI24 element with small displacements.
epsi26ns1.inp	PSI26 element with isotropic behavior.
epsi26nn2.inp	PSI26 element with temperature dependence, large displacements.
epsi34np1.inp	PSI34 element with perturbation.
epsi34ns1.inp	PSI34 element with temperature dependence, small displacements.
epsi36ns1.inp	PSI36 element with large displacements.

ASCE formulae for sand:

epsi24ss1.inp	PSI24 element with small displacements.
epsi24sn1.inp	PSI24 element with large displacements, user-defined orientation.
epsi34sn1.inp	PSI34 element with temperature dependence, large displacements.
epsi36sn1.inp	PSI36 element with large displacements, temperature dependence.

ASCE formulae for clay:

epsi24cn1.inp	PSI24 element with large displacements, user-defined orientation.
epsi26cn1.inp	PSI26 element with large displacements, user-defined orientation.
epsi34cs1.inp	PSI34 element with small displacements.
epsi34cn3.inp	PSI34 element with field variable dependence, large displacements.

User subroutine:

epsi26un1.inp	PSI26 element with large displacements.
epsi26un1.f	The user subroutine used with epsi26un1.inp.
epsi34us1.inp	PSI34 element with small displacements.
epsi34us1.f	The user subroutine used with epsi34us1.inp.

Shear panel elements

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

SHEAR4

Problem description

In this problem a state of simple shear is induced in a single element up to a nominal shear strain of 2%. The material model is isotropic linear elasticity.

The material properties used are Young's modulus = 1.0×10^7 and Poisson's ratio = 0.3.

Results and discussion

The computed equivalent shear flow and reaction forces agree well with analytic solutions.

Input files

shear_shear4.inp	SHEAR4 element.

Element loading options

In this section:

- *Continuum stress/displacement elements*
- *Beam stress/displacement elements*
- *Pipe stress/displacement elements*
- *Shell, membrane, and truss stress/displacement elements*
- *Cohesive element load verification*
- *ELBOW elements*
- *Continuum pore pressure elements*
- *Continuum and shell heat transfer elements*
- *Coupled temperature-displacement elements*
- *Coupled thermal-electrical-structural elements*
- *Piezoelectric elements*
- *Continuum mass diffusion elements*
- *Thermal-electrical elements*
- *Rigid elements*
- *Mass and rotary inertia elements*
- *Abaqus/Explicit element loading verification*
- *Surface tension loads*
- *Incident wave loading*
- *Distributed traction and edge loads*

Continuum stress/displacement elements

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Plane stress, plane strain, and generalized plane strain elements

Problem description



Note:

Meshes for tests of foundation types F_2, F_3, and F_4 are irregularly shaped.

Model:

Square dimensions	7×7
Thickness	1.0
Centrifugal axis of rotation	(0, 1, 0) through origin
Coriolis axis of rotation	(0, 0, 1) through origin
Gravitational load vector	(0, −1, 0)

Material:

Young's modulus	3×10^6
Poisson's ratio	0.3
Coefficient of thermal expansion	.0001
Density	5×10^{-5}

Initial conditions:

Initial temperature	ALL, −10.0
Initial velocity	ALL, 1, 10.0
(Coriolis loading)	ALL, 2, 5.0
Hydrostatic pressure datum	lower-order elements: 7.0
	higher-order elements: 3.0
Hydrostatic pressure elevation	0.0

Results and discussion

The calculated reactions are in agreement with the applied loads.

Input files

CPEG3 element load tests:

ecg3sfd1.inp	BX, BY, CENT, CENTRIF, GRAV, HP1, HP2, HP3, P1, P2, P3, **TEMPERATURE*.
ecg3sfd2.inp	F1.
ecg3sfd3.inp	F2.
ecg3sfd4.inp	F3.
ecg3sfda.inp	CORIO.
ecg3sfdc.inp	CORIO.
ecg3sfdi.inp	HP, P, **TEMPERATURE*.
ecg3sfdr.inp	ROTA.

CPEG3H element load tests:

ecg3shd1.inp	BX, BY, CENT, CENTRIF, GRAV, HP1, HP2, HP3, P1, P2, P3, **TEMPERATURE*.
ecg3shd2.inp	F1.
ecg3shd3.inp	F2.
ecg3shd4.inp	F3.
ecg3shda.inp	CORIO.
ecg3shdi.inp	HP, P, **TEMPERATURE*.
ecg3shdr.inp	ROTA.

CPEG4 element load tests:

ecg4sfd1.inp	BX, BY, CENT, CENTRIF, GRAV, HP1, HP2, HP3, HP4, P1, P2, P3, P4, **TEMPERATURE*.
ecg4sfd2.inp	F1.
ecg4sfd3.inp	F2.
ecg4sfd4.inp	F3.
ecg4sfd5.inp	F4.
ecg4sfda.inp	CORIO.
ecg4sfdc.inp	CORIO.
ecg4sfdi.inp	HP, P, **TEMPERATURE*.
ecg4sfdr.inp	ROTA.

CPEG4H element load tests:

ecg4shd1.inp	BX, BY, CENT, CENTRIF, GRAV, HP1, HP2, HP3, HP4, P1, P2, P3, P4, **TEMPERATURE*.
ecg4shd2.inp	F1.
ecg4shd3.inp	F2.
ecg4shd4.inp	F3.
ecg4shd5.inp	F4.
ecg4shda.inp	CORIO.
ecg4shdi.inp	HP, P, **TEMPERATURE*.
ecg4shdr.inp	ROTA.

CPEG4I element load tests:

ecg4sid1.inp	BX, BY, CENT, CENTRIF, GRAV, HP1, HP2, HP3, HP4, P1, P2, P3, P4, **TEMPERATURE*.
ecg4sid2.inp	F1.
ecg4sid3.inp	F2.
ecg4sid4.inp	F3.
ecg4sid5.inp	F4.
ecg4sida.inp	CORIO.
ecg4sidi.inp	HP, P, **TEMPERATURE*.
ecg4sidr.inp	ROTA.

CPEG4IH element load tests:

ecg4sjd1.inp	BX, BY, CENT, CENTRIF, GRAV, HP1, HP2, HP3, HP4, P1, P2, P3, P4, **TEMPERATURE*.
ecg4sjd2.inp	F1.
ecg4sjd3.inp	F2.
ecg4sjd4.inp	F3.
ecg4sjd5.inp	F4.
ecg4sjda.inp	CORIO.
ecg4sjdi.inp	HP, P, **TEMPERATURE*.
ecg4sjdr.inp	ROTA.

CPEG4R element load tests:

ecg4srd1.inp	BX, BY, CENT, CENTRIF, GRAV, HP1, HP2, HP3, HP4, P1, P2, P3, P4, **TEMPERATURE*.
ecg4srd2.inp	F1.
ecg4srd3.inp	F2.
ecg4srd4.inp	F3.
ecg4srd5.inp	F4.
ecg4srda.inp	CORIO.
ecg4srdi.inp	HP, P, **TEMPERATURE*.
ecg4srdr.inp	ROTA.

CPEG4RH element load tests:

ecg4syd1.inp	BX, BY, CENT, CENTRIF, GRAV, HP1, HP2, HP3, HP4, P1, P2, P3, P4, **TEMPERATURE*.
ecg4syd2.inp	F1.
ecg4syd3.inp	F2.
ecg4syd4.inp	F3.
ecg4syd5.inp	F4.
ecg4syda.inp	CORIO.
ecg4sydi.inp	HP, P, **TEMPERATURE*.
ecg4sydr.inp	ROTA.

CPEG6 element load tests:

ecg6sfd1.inp	BX, BY, CENT, CENTRIF, GRAV, HP1, HP2, HP3, P1, P2, P3, *TEMPERATURE.
ecg6sfd2.inp	F1.
ecg6sfd3.inp	F2.
ecg6sfd4.inp	F3.
ecg6sfda.inp	CORIO.
ecg6sfdc.inp	CORIO.
ecg6sfdi.inp	HP, P, *TEMPERATURE.
ecg6sfdr.inp	ROTA.

CPEG6H element load tests:

ecg6shd1.inp	BX, BY, CENT, CENTRIF, GRAV, HP1, HP2, HP3, P1, P2, P3, *TEMPERATURE.
ecg6shd2.inp	F1.
ecg6shd3.inp	F2.
ecg6shd4.inp	F3.
ecg6shda.inp	CORIO.
ecg6shdi.inp	HP, P, *TEMPERATURE.
ecg6shdr.inp	ROTA.

CPEG6M element load tests:

ecg6skd1.inp	BX, BY, CENT, CENTRIF, GRAV, HP1, HP2, HP3, P1, P2, P3, *TEMPERATURE.
ecg6skd2.inp	F1.
ecg6skd3.inp	F2.
ecg6skd4.inp	F3.
ecg6skda.inp	CORIO.
ecg6skdc.inp	CORIO.
ecg6skdi.inp	HP, P, *TEMPERATURE.
ecg6skdr.inp	ROTA.

CPEG6MH element load tests:

ecg6sld1.inp	BX, BY, CENT, CENTRIF, GRAV, HP1, HP2, HP3, P1, P2, P3, *TEMPERATURE.
ecg6sld2.inp	F1.
ecg6sld3.inp	F2.
ecg6sld4.inp	F3.
ecg6slda.inp	CORIO.
ecg6sldi.inp	HP, P, *TEMPERATURE.
ecg6sldr.inp	ROTA.

CPEG8 element load tests:

ecg8sfd1.inp	BX, BY, CENT, CENTRIF, GRAV, HP1, HP2, HP3, HP4, P1, P2, P3, P4, *TEMPERATURE.
ecg8sfd2.inp	F1.
ecg8sfd3.inp	F2.

ecg8sfd4.inp	F3.
ecg8sfd5.inp	F4.
ecg8sfda.inp	CORIO.
ecg8sfdc.inp	CORIO.
ecg8sfdi.inp	HP, P, **TEMPERATURE*.
ecg8sfdr.inp	ROTA.
CPEG8H element load tests:	
ecg8shd1.inp	BX, BY, CENT, CENTRIF, GRAV, HP1, HP2, HP3, HP4, P1, P2, P3, P4, **TEMPERATURE*.
ecg8shd2.inp	F1.
ecg8shd3.inp	F2.
ecg8shd4.inp	F3.
ecg8shd5.inp	F4.
ecg8shda.inp	CORIO.
ecg8shdi.inp	HP, P, **TEMPERATURE*.
ecg8shdr.inp	ROTA.
CPEG8R element load tests:	
ecg8srd1.inp	BX, BY, CENT, CENTRIF, GRAV, HP1, HP2, HP3, HP4, P1, P2, P3, P4, **TEMPERATURE*.
ecg8srd2.inp	F1.
ecg8srd3.inp	F2.
ecg8srd4.inp	F3.
ecg8srd5.inp	F4.
ecg8srda.inp	CORIO.
ecg8srdi.inp	HP, P, **TEMPERATURE*.
ecg8srdr.inp	ROTA.
CPEG8RH element load tests:	
ecg8syd1.inp	BX, BY, CENT, CENTRIF, GRAV, HP1, HP2, HP3, HP4, P1, P2, P3, P4, **TEMPERATURE*.
ecg8syd2.inp	F1.
ecg8syd3.inp	F2.
ecg8syd4.inp	F3.
ecg8syd5.inp	F4.
ecg8syda.inp	CORIO.
ecg8sydi.inp	HP, P, **TEMPERATURE*.
ecg8sydr.inp	ROTA.
CPE3 element load tests:	
ece3sfd1.inp	BX, BY, CENT, CENTRIF, GRAV, HP1, HP2, HP3, P1, P2, P3, **TEMPERATURE*.
ece3sfd2.inp	F1.
ece3sfd3.inp	F2.
ece3sfd4.inp	F3.

ece3sfda.inp	CORIO.
ece3sfdc.inp	CORIO.
ece3sfdi.inp	HP, P, **TEMPERATURE*.
ece3sfdr.inp	ROTA.

CPE3H element load tests:

ece3shd1.inp	BX, BY, CENT, CENTRIF, GRAV, HP1, HP2, HP3, P1, P2, P3, **TEMPERATURE*.
ece3shd2.inp	F1.
ece3shd3.inp	F2.
ece3shd4.inp	F3.
ece3shda.inp	CORIO.
ece3shdi.inp	HP, P, **TEMPERATURE*.
ece3shdr.inp	ROTA.

CPE4 element load tests:

ece4sfd1.inp	BX, BY, CENT, CENTRIF, GRAV, HP1, HP2, HP3, HP4, P1, P2, P3, P4, **TEMPERATURE*.
ece4sfd2.inp	F1.
ece4sfd3.inp	F2.
ece4sfd4.inp	F3.
ece4sfd5.inp	F4.
ece4sfda.inp	CORIO.
ece4sfdc.inp	CORIO.
ece4sfdg.inp	CORIO.
ece4sfdi.inp	HP, P, **TEMPERATURE*.
ece4sfdr.inp	ROTA.
ece4sfds.inp	CORIO.

CPE4H element load tests:

ece4shd1.inp	BX, BY, CENT, CENTRIF, GRAV, HP1, HP2, HP3, HP4, P1, P2, P3, P4, **TEMPERATURE*.
ece4shd2.inp	F1.
ece4shd3.inp	F2.
ece4shd4.inp	F3.
ece4shd5.inp	F4.
ece4shda.inp	CORIO.
ece4shdi.inp	HP, P, **TEMPERATURE*.
ece4shdr.inp	ROTA.

CPE4I element load tests:

ece4sid1.inp	BX, BY, CENT, CENTRIF, GRAV, HP1, HP2, HP3, HP4, P1, P2, P3, P4, **TEMPERATURE*.
ece4sid2.inp	F1.
ece4sid3.inp	F2.
ece4sid4.inp	F3.

ece4sid5.inp	F4.
ece4sida.inp	CORIO.
ece4sidi.inp	HP, P, **TEMPERATURE*.
ece4sidr.inp	ROTA.

CPE4IH element load tests:

ece4sjd1.inp	BX, BY, CENT, CENTRIF, GRAV, HP1, HP2, HP3, HP4, P1, P2, P3, P4, **TEMPERATURE*.
ece4sjd2.inp	F1.
ece4sjd3.inp	F2.
ece4sjd4.inp	F3.
ece4sjd5.inp	F4.
ece4sjda.inp	CORIO.
ece4sjdi.inp	HP, P, **TEMPERATURE*.
ece4sjdr.inp	ROTA.

CPE4R element load tests:

ece4srd1.inp	BX, BY, CENT, CENTRIF, GRAV, HP1, HP2, HP3, HP4, P1, P2, P3, P4, **TEMPERATURE*.
ece4srd2.inp	F1.
ece4srd3.inp	F2.
ece4srd4.inp	F3.
ece4srd5.inp	F4.
ece4srda.inp	CORIO.
ece4srdi.inp	HP, P, **TEMPERATURE*.
ece4srdr.inp	ROTA.

CPE4RH element load tests:

ece4syd1.inp	BX, BY, CENT, CENTRIF, GRAV, HP1, HP2, HP3, HP4, P1, P2, P3, P4, **TEMPERATURE*.
ece4syd2.inp	F1.
ece4syd3.inp	F2.
ece4syd4.inp	F3.
ece4syd5.inp	F4.
ece4syda.inp	CORIO.
ece4sydi.inp	HP, P, **TEMPERATURE*.
ece4sydr.inp	ROTA.

CPE6 element load tests:

ece6sfd1.inp	BX, BY, CENT, CENTRIF, GRAV, HP1, HP2, HP3, P1, P2, P3, **TEMPERATURE*.
ece6sfd2.inp	F1.
ece6sfd3.inp	F2.
ece6sfd4.inp	F3.
ece6sfda.inp	CORIO.
ece6sfdc.inp	CORIO.

ece6sfdr.inp	ROTA.
CPE6H element load tests:	
ece6shd1.inp	BX, BY, CENT, CENTRIF, GRAV, HP1, HP2, HP3, P1, P2, P3, **TEMPERATURE*.
ece6shd2.inp	F1.
ece6shd3.inp	F2.
ece6shd4.inp	F3.
ece6shda.inp	CORIO.
ece6shdr.inp	ROTA.
CPE6M element load tests:	
ece6skd1.inp	BX, BY, CENT, CENTRIF, GRAV, HP1, HP2, HP3, P1, P2, P3, **TEMPERATURE*.
ece6skd2.inp	F1.
ece6skd3.inp	F2.
ece6skd4.inp	F3.
ece6skda.inp	CORIO.
ece6skdi.inp	HP, P, **TEMPERATURE*.
ece6skdr.inp	ROTA.
CPE6MH element load tests:	
ece6sld1.inp	BX, BY, CENT, CENTRIF, GRAV, HP1, HP2, HP3, P1, P2, P3, **TEMPERATURE*.
ece6sld2.inp	F1.
ece6sld3.inp	F2.
ece6sld4.inp	F3.
ece6slda.inp	CORIO.
ece6sldi.inp	HP, P, **TEMPERATURE*.
ece6sldr.inp	ROTA.
CPE8 element load tests:	
ece8sfd1.inp	BX, BY, CENT, CENTRIF, GRAV, HP1, HP2, HP3, HP4, P1, P2, P3, P4, **TEMPERATURE*.
ece8sfd2.inp	F1.
ece8sfd3.inp	F2.
ece8sfd4.inp	F3.
ece8sfd5.inp	F4.
ece8sfda.inp	CORIO.
ece8sfdc.inp	CORIO.
ece8sfdi.inp	HP, P, **TEMPERATURE*.
ece8sfdr.inp	ROTA.
CPE8H element load tests:	
ece8shd1.inp	BX, BY, CENT, CENTRIF, GRAV, HP1, HP2, HP3, HP4, P1, P2, P3, P4, **TEMPERATURE*.
ece8shd2.inp	F1.

ece8shd3.inp	F2.
ece8shd4.inp	F3.
ece8shd5.inp	F4.
ece8shda.inp	CORIO.
ece8shdi.inp	HP, P, **TEMPERATURE*.
ece8shdr.inp	ROTA.

CPE8R element load tests:

ece8srd1.inp	BX, BY, CENT, CENTRIF, GRAV, HP1, HP2, HP3, HP4, P1, P2, P3, P4, **TEMPERATURE*.
ece8srd2.inp	F1.
ece8srd3.inp	F2.
ece8srd4.inp	F3.
ece8srd5.inp	F4.
ece8srda.inp	CORIO.
ece8srdi.inp	HP, P, **TEMPERATURE*.
ece8srdr.inp	ROTA.

CPE8RH element load tests:

ece8syd1.inp	BX, BY, CENT, CENTRIF, GRAV, HP1, HP2, HP3, HP4, P1, P2, P3, P4, **TEMPERATURE*.
ece8syd2.inp	F1.
ece8syd3.inp	F2.
ece8syd4.inp	F3.
ece8syd5.inp	F4.
ece8syda.inp	CORIO.
ece8sydi.inp	HP, P, **TEMPERATURE*.
ece8sydr.inp	ROTA.

CPS3 element load tests:

ecs3sfd1.inp	BX, BY, CENT, CENTRIF, GRAV, HP1, HP2, HP3, P1, P2, P3, **TEMPERATURE*.
ecs3sfd2.inp	F1.
ecs3sfd3.inp	F2.
ecs3sfd4.inp	F3.
ecs3sfda.inp	CORIO.
ecs3sfdc.inp	CORIO.
ecs3sfdi.inp	HP, P, **TEMPERATURE*.

CPS4 element load tests:

ecs4sfd1.inp	BX, BY, CENT, CENTRIF, GRAV, HP1, HP2, HP3, HP4, P1, P2, P3, P4, **TEMPERATURE*.
ecs4sfd2.inp	F1.
ecs4sfd3.inp	F2.
ecs4sfd4.inp	F3.
ecs4sfd5.inp	F4.

ecs4sfda.inp	CORIO.
ecs4sfdc.inp	CORIO.
ecs4sfdi.inp	HP, P, **TEMPERATURE*.
ecs4sfdr.inp	ROTA.

CPS4I element load tests:

ecs4sid1.inp	BX, BY, CENT, CENTRIF, GRAV, HP1, HP2, HP3, HP4, P1, P2, P3, P4, **TEMPERATURE*.
ecs4sid2.inp	F1.
ecs4sid3.inp	F2.
ecs4sid4.inp	F3.
ecs4sid5.inp	F4.
ecs4sida.inp	CORIO.
ecs4sidi.inp	HP, P, **TEMPERATURE*.
ecs4sidr.inp	ROTA.

CPS4R element load tests:

ecs4srd1.inp	BX, BY, CENT, CENTRIF, GRAV, HP1, HP2, HP3, HP4, P1, P2, P3, P4, **TEMPERATURE*.
ecs4srd2.inp	F1.
ecs4srd3.inp	F2.
ecs4srd4.inp	F3.
ecs4srd5.inp	F4.
ecs4srda.inp	CORIO.
ecs4srdi.inp	HP, P, **TEMPERATURE*.

CPS6 element load tests:

ecs6sfd1.inp	BX, BY, CENT, CENTRIF, GRAV, HP1, HP2, HP3, P1, P2, P3, **TEMPERATURE*.
ecs6sfd2.inp	F1.
ecs6sfd3.inp	F2.
ecs6sfd4.inp	F3.
ecs6sfda.inp	CORIO.
ecs6sfdc.inp	CORIO.
ecs6sfdi.inp	HP, P, **TEMPERATURE*.
ecs6sfdr.inp	ROTA.

CPS6M element load tests:

ecs6skd1.inp	BX, BY, CENT, CENTRIF, GRAV, HP1, HP2, HP3, P1, P2, P3, **TEMPERATURE*.
ecs6skd2.inp	F1.
ecs6skd3.inp	F2.
ecs6skd4.inp	F3.
ecs6skda.inp	CORIO.
ecs6skdi.inp	HP, P, **TEMPERATURE*.
ecs6skdr.inp	ROTA.

CPS8 element load tests:

ecs8sfd1.inp	BX, BY, CENT, CENTRIF, GRAV, HP1, HP2, HP3, HP4, P1, P2, P3, P4, **TEMPERATURE*.
ecs8sfd2.inp	F1.
ecs8sfd3.inp	F2.
ecs8sfd4.inp	F3.
ecs8sfd5.inp	F4.
ecs8sfda.inp	CORIO.
ecs8sfdc.inp	CORIO.
ecs8sfdi.inp	HP, P, **TEMPERATURE*.
ecs8sfdr.inp	ROTA.

CPS8R element load tests:

ecs8srd1.inp	BX, BY, CENT, CENTRIF, GRAV, HP1, HP2, HP3, HP4, P1, P2, P3, P4, **TEMPERATURE*.
ecs8srd2.inp	F1.
ecs8srd3.inp	F2.
ecs8srd4.inp	F3.
ecs8srd5.inp	F4.
ecs8srda.inp	CORIO.
ecs8srdi.inp	HP, P, **TEMPERATURE*.
ecs8srdr.inp	ROTA.

Axisymmetric solid elements

Problem description

Model:

Planar dimensions	3×3
Centrifugal axis of rotation	(0, 1, 0) through origin
Gravitational load vector	(0, −1, 0)

Material:

Young's modulus	3×10^6
Poisson's ratio	0.3
Density	5×10^{-5}

Initial conditions:

Hydrostatic pressure datum	3.0
Hydrostatic pressure elevation	0.0

Results and discussion

The calculated reactions are in agreement with the applied loads.

Input files

CAX3 element load tests:

eca3sfd1.inp	BR, BZ, CENT, CENTRIF, GRAV, HP1, HP2, HP3, P1, P2, P3.
eca3sfd2.inp	F1.
eca3sfd3.inp	F2.
eca3sfd4.inp	F3.
eca3sfdi.inp	HP, P.

CAX3H element load tests:

eca3shd1.inp	BR, BZ, CENT, CENTRIF, GRAV, HP1, HP2, HP3, P1, P2, P3.
eca3shd2.inp	F1.
eca3shd3.inp	F2.
eca3shd4.inp	F3.

CAX4 element load tests:

eca4sfd1.inp	BR, BZ, CENT, CENTRIF, GRAV, HP1, HP2, HP3, HP4, P1, P2, P3, P4.
eca4sfd2.inp	F1.
eca4sfd3.inp	F2.
eca4sfd4.inp	F3.
eca4sfd5.inp	F4.
eca4sfdi.inp	HP, P.

CAX4H element load tests:

eca4shd1.inp	BR, BZ, CENT, CENTRIF, GRAV, HP1, HP2, HP3, HP4, P1, P2, P3, P4.
eca4shd2.inp	F1.
eca4shd3.inp	F2.
eca4shd4.inp	F3.
eca4shd5.inp	F4.
eca4shdi.inp	HP, P.

CAX4I element load tests:

eca4sid1.inp	BR, BZ, CENT, CENTRIF, GRAV, HP1, HP2, HP3, HP4, P1, P2, P3, P4.
eca4sid2.inp	F1.
eca4sid3.inp	F2.
eca4sid4.inp	F3.
eca4sid5.inp	F4.
eca4sidi.inp	HP, P.

CAX4IH element load tests:

eca4sjd1.inp	BR, BZ, CENT, CENTRIF, GRAV, HP1, HP2, HP3, HP4, P1, P2, P3, P4.
eca4sjd2.inp	F1.
eca4sjd3.inp	F2.
eca4sjd4.inp	F3.
eca4sjd5.inp	F4.
eca4sjdi.inp	HP, P.

CAX4R element load tests:

eca4srd1.inp	BR, BZ, CENT, CENTRIF, GRAV, HP1, HP2, HP3, HP4, P1, P2, P3, P4.
eca4srd2.inp	F1.
eca4srd3.inp	F2.
eca4srd4.inp	F3.
eca4srd5.inp	F4.
eca4srdi.inp	HP, P.

CAX4RH element load tests:

eca4syd1.inp	BR, BZ, CENT, CENTRIF, GRAV, HP1, HP2, HP3, HP4, P1, P2, P3, P4.
eca4syd2.inp	F1.
eca4syd3.inp	F2.
eca4syd4.inp	F3.
eca4syd5.inp	F4.
eca4sydi.inp	HP, P.

CAX6 element load tests:

eca6sfd1.inp	BR, BZ, CENT, CENTRIF, GRAV, HP1, HP2, HP3, P1, P2, P3.
eca6sfd2.inp	F1.
eca6sfd3.inp	F2.
eca6sfd4.inp	F3.
eca6sfdi.inp	HP, P.

CAX6H element load tests:

eca6shd1.inp	BR, BZ, CENT, CENTRIF, GRAV, HP1, HP2, HP3, P1, P2, P3.
eca6shd2.inp	F1.
eca6shd3.inp	F2.
eca6shd4.inp	F3.
eca6shdi.inp	HP, P.

CAX6M element load tests:

eca6skd1.inp	BR, BZ, CENT, CENTRIF, GRAV, HP1, HP2, HP3, P1, P2, P3.
eca6skd2.inp	F1.

eca6skd3.inp	F2.
eca6skd4.inp	F3.
eca6skdi.inp	HP, P.

CAX6MH element load tests:

eca6sld1.inp	BR, BZ, CENT, CENTRIF, GRAV, HP1, HP2, HP3, P1, P2, P3.
eca6sld2.inp	F1.
eca6sld3.inp	F2.
eca6sld4.inp	F3.
eca6sldi.inp	HP, P.

CAX8 element load tests:

eca8sfd1.inp	BR, BZ, CENT, CENTRIF, GRAV, HP1, HP2, HP3, HP4, P1, P2, P3, P4.
eca8sfd2.inp	F1.
eca8sfd3.inp	F2.
eca8sfd4.inp	F3.
eca8sfd5.inp	F4.
eca8sfdi.inp	HP, P.

CAX8H element load tests:

eca8shd1.inp	BR, BZ, CENT, CENTRIF, GRAV, HP1, HP2, HP3, HP4, P1, P2, P3, P4.
eca8shd2.inp	F1.
eca8shd3.inp	F2.
eca8shd4.inp	F3.
eca8shd5.inp	F4.
eca8shdi.inp	HP, P.

CAX8R element load tests:

eca8srd1.inp	BR, BZ, CENT, CENTRIF, GRAV, HP1, HP2, HP3, HP4, P1, P2, P3, P4.
eca8srd2.inp	F1.
eca8srd3.inp	F2.
eca8srd4.inp	F3.
eca8srd5.inp	F4.
eca8srdi.inp	HP, P.

CAX8RH element load tests:

eca8syd1.inp	BR, BZ, CENT, CENTRIF, GRAV, HP1, HP2, HP3, HP4, P1, P2, P3, P4.
eca8syd2.inp	F1.
eca8syd3.inp	F2.
eca8syd4.inp	F3.
eca8syd5.inp	F4.

Three-dimensional solids

Problem description

Model:

Cubic dimensions	$7 \times 7 \times 7$
Centrifugal and Coriolis axes of rotation	(0, 1, 0) through (−1000, 3.5, −3.5)
Gravitational load vector	(1, 0, 0)

Material:

Young's modulus	3×10^6
Poisson's ratio	0.3
Coefficient of thermal expansion	.0001
Density	10.0

Initial conditions:

Initial temperature	ALL, −10.0
Initial velocity	ALL, 1, 10.0
(Coriolis loading)	ALL, 2, 5.0
Hydrostatic pressure datum	0.0
Hydrostatic pressure elevation	−7.0

Results and discussion

The calculated reactions are in agreement with the applied loads.

Input files

C3D4 element load tests:

ec34sfd1.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, P1, P2, P3, P4, HP1, HP2, HP3, HP4, **TEMPERATURE*.
ec34sfd2.inp	F1.
ec34sfd3.inp	F2.
ec34sfd4.inp	F3.
ec34sfd5.inp	F4.
ec34sfda.inp	CORIO.
ec34sfdc.inp	CORIO.
ec34sfdi.inp	HP, P, **TEMPERATURE*.
ec34sfdr.inp	ROTA.

C3D4H element load tests:

ec34shd1.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, P1, P2, P3, P4, HP1, HP2, HP3, HP4, **TEMPERATURE*.

ec34shd2.inp	F1.
ec34shd3.inp	F2.
ec34shd4.inp	F3.
ec34shd5.inp	F4.
ec34shda.inp	CORIO.
ec34shdr.inp	ROTA.

C3D5 element load tests:

ec35sfd1.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, P1, P2, P3, P4, P5, HP1, HP2, HP3, HP4, HP5, **TEMPERATURE*.
ec35sfd2.inp	F1.
ec35sfd3.inp	F2.
ec35sfd4.inp	F3.
ec35sfd5.inp	F4 and F5.
ec35sfda.inp	CORIO.
ec35sfdc.inp	CORIO.
ec35sfdi.inp	HP, P, **TEMPERATURE*.
ec35sfdr.inp	ROTA.

C3D5H element load tests:

ec35shd1.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, P1, P2, P3, P4, P5, HP1, HP2, HP3, HP4, HP5, **TEMPERATURE*.
ec35shd2.inp	F1.
ec35shd3.inp	F2.
ec35shd4.inp	F3.
ec35shd5.inp	F4 and F5.
ec35shda.inp	CORIO.
ec35shdc.inp	CORIO.
ec35shdi.inp	HP, P, **TEMPERATURE*.
ec35shdr.inp	ROTA.

C3D6 element load tests:

ec36sfd1.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, P1, P2, P3, P4, P5, HP1, HP2, HP3, HP4, HP5, **TEMPERATURE*.
ec36sfd2.inp	F1.
ec36sfd3.inp	F2.
ec36sfd4.inp	F3.
ec36sfd5.inp	F4.
ec36sfd6.inp	F5.
ec36sfda.inp	CORIO.
ec36sfdc.inp	CORIO.
ec36sfdi.inp	HP, P, **TEMPERATURE*.
ec36sfdr.inp	ROTA.

C3D6H element load tests:

ec36shd1.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, P1, P2, P3, P4, P5, HP1, HP2, HP3, HP4, HP5, **TEMPERATURE*.
ec36shd2.inp	F1.
ec36shd3.inp	F2.
ec36shd4.inp	F3.
ec36shd5.inp	F4.
ec36shd6.inp	F5.
ec36shda.inp	CORIO.
ec36shdi.inp	HP, P, **TEMPERATURE*.
ec36shdr.inp	ROTA.

C3D8 element load tests:

ec38sfd1.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, P1, P2, P3, P4, P5, P6, HP1, HP2, HP3, HP4, HP5, HP6, **TEMPERATURE*.
ec38sfd2.inp	F1.
ec38sfd3.inp	F2.
ec38sfd4.inp	F3.
ec38sfd5.inp	F4.
ec38sfd6.inp	F5.
ec38sfd7.inp	F6.
ec38sfda.inp	CORIO.
ec38sfdc.inp	CORIO.
ec38sfdi.inp	HP, P, **TEMPERATURE*.
ec38sfdr.inp	ROTA.

C3D8S element load tests:

c3d8s_loads.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, P1, P2, P3, P4, P5, P6, HP1, HP2, HP3, HP4, HP5, HP6, **TEMPERATURE*.
c3d8s_dsloads.inp	HP, P, **TEMPERATURE*.

C3D8HS element load tests:

c3d8hs_loads.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, P1, P2, P3, P4, P5, P6, HP1, HP2, HP3, HP4, HP5, HP6, **TEMPERATURE*.
c3d8hs_dsloads.inp	HP, P, **TEMPERATURE*.

C3D8H element load tests:

ec38shd1.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, P1, P2, P3, P4, P5, P6, HP1, HP2, HP3, HP4, HP5, HP6, **TEMPERATURE*.
ec38shd2.inp	F1.
ec38shd3.inp	F2.
ec38shd4.inp	F3.
ec38shd5.inp	F4.
ec38shd6.inp	F5.
ec38shd7.inp	F6.
ec38shda.inp	CORIO.
ec38shdi.inp	HP, P, **TEMPERATURE*.

ec38shdr.inp	ROTA.
C3D8I element load tests:	
ec38sid1.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, P1, P2, P3, P4, P5, P6, HP1, HP2, HP3, HP4, HP5, HP6, **TEMPERATURE*.
ec38sid2.inp	F1.
ec38sid3.inp	F2.
ec38sid4.inp	F3.
ec38sid5.inp	F4.
ec38sid6.inp	F5.
ec38sid7.inp	F6.
ec38sida.inp	CORIO.
ec38sidi.inp	HP, P, **TEMPERATURE*.
ec38sidr.inp	ROTA.
C3D8IH element load tests:	
ec38sjd1.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, P1, P2, P3, P4, P5, P6, HP1, HP2, HP3, HP4, HP5, HP6, **TEMPERATURE*.
ec38sjd2.inp	F1.
ec38sjd3.inp	F2.
ec38sjd4.inp	F3.
ec38sjd5.inp	F4.
ec38sjd6.inp	F5.
ec38sjd7.inp	F6.
ec38sjda.inp	CORIO.
ec38sjdi.inp	HP, P, **TEMPERATURE*.
ec38sjdr.inp	ROTA.
C3D8R element load tests:	
ec38srd1.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, P1, P2, P3, P4, P5, P6, HP1, HP2, HP3, HP4, HP5, HP6, **TEMPERATURE*.
ec38srd2.inp	F1.
ec38srd3.inp	F2.
ec38srd4.inp	F3.
ec38srd5.inp	F4.
ec38srd6.inp	F5.
ec38srd7.inp	F6.
ec38srda.inp	CORIO.
ec38srdi.inp	HP, P, **TEMPERATURE*.
ec38srdr.inp	ROTA.
C3D8RH element load tests:	
ec38syd1.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, P1, P2, P3, P4, P5, P6, HP1, HP2, HP3, HP4, HP5, HP6, **TEMPERATURE*.
ec38syd2.inp	F1.
ec38syd3.inp	F2.

ec38syd4.inp	F3.
ec38syd5.inp	F4.
ec38syd6.inp	F5.
ec38syd7.inp	F6.
ec38syda.inp	CORIO.
ec38sydi.inp	HP, P, **TEMPERATURE*.
ec38sydr.inp	ROTA.
C3D10 element load tests:	
ec3asfd1.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, P1, P2, P3, P4, HP1, HP2, HP3, HP4, **TEMPERATURE*.
ec3asfd2.inp	F1.
ec3asfd3.inp	F2.
ec3asfd4.inp	F3.
ec3asfd5.inp	F4.
ec3asfda.inp	CORIO.
ec3asfdc.inp	CORIO.
ec3asfdr.inp	ROTA.
C3D10H element load tests:	
ec3ashd1.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, P1, P2, P3, P4, HP1, HP2, HP3, HP4, **TEMPERATURE*.
ec3ashd2.inp	F1.
ec3ashd3.inp	F2.
ec3ashd4.inp	F3.
ec3ashd5.inp	F4.
ec3ashda.inp	CORIO.
ec3ashdr.inp	ROTA.
C3D10HS element load tests:	
ec3asid1.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, P1, P2, P3, P4, HP1, HP2, HP3, HP4, **TEMPERATURE*.
ec3asid2.inp	F1.
ec3asid3.inp	F2.
ec3asid4.inp	F3.
ec3asid5.inp	F4.
ec3asida.inp	CORIO.
ec3asidc.inp	CORIO.
ec3asidr.inp	ROTA.
C3D10M element load tests:	
ec3askd1.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, P1, P2, P3, P4, HP1, HP2, HP3, HP4, **TEMPERATURE*.
ec3askd2.inp	F1.
ec3askd3.inp	F2.
ec3askd4.inp	F3.

ec3askd5.inp	F4.
ec3askda.inp	CORIO.
ec3askdr.inp	ROTA.

C3D10MH element load tests:

ec3asld1.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, P1, P2, P3, P4, HP1, HP2, HP3, HP4, **TEMPERATURE*.
ec3asld2.inp	F1.
ec3asld3.inp	F2.
ec3asld4.inp	F3.
ec3asld5.inp	F4.
ec3aslda.inp	CORIO.
ec3asldr.inp	ROTA.

C3D15 element load tests:

ec3fsfd1.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, P1, P2, P3, P4, P5, HP1, HP2, HP3, HP4, HP5, **TEMPERATURE*.
ec3fsfd2.inp	F1.
ec3fsfd3.inp	F2.
ec3fsfd4.inp	F3.
ec3fsfd5.inp	F4.
ec3fsfd6.inp	F5.
ec3fsfda.inp	CORIO.
ec3fsfdc.inp	CORIO.
ec3fsfdi.inp	HP, P, **TEMPERATURE*.
ec3fsfdr.inp	ROTA.

C3D15H element load tests:

ec3fshd1.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, P1, P2, P3, P4, P5, HP1, HP2, HP3, HP4, HP5, **TEMPERATURE*.
ec3fshd2.inp	F1.
ec3fshd3.inp	F2.
ec3fshd4.inp	F3.
ec3fshd5.inp	F4.
ec3fshd6.inp	F5.
ec3fshda.inp	CORIO.
ec3fshdi.inp	HP, P, **TEMPERATURE*.
ec3fshdr.inp	ROTA.

C3D20 element tests:

ec3ksfd1.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, P1, P2, P3, P4, P5, P6, HP1, HP2, HP3, HP4, HP5, HP6, **TEMPERATURE*.
ec3ksfd2.inp	F1.
ec3ksfd3.inp	F2.
ec3ksfd4.inp	F3.
ec3ksfd5.inp	F4.

ec3ksfd6.inp	F5.
ec3ksfd7.inp	F6.
ec3ksfda.inp	CORIO.
ec3ksfdc.inp	CORIO.
ec3ksfdi.inp	HP, P, **TEMPERATURE*.
ec3ksfdr.inp	ROTA.
C3D20H element load tests:	
ec3kshd1.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, P1, P2, P3, P4, P5, P6, HP1, HP2, HP3, HP4, HP5, HP6, **TEMPERATURE*.
ec3kshd2.inp	F1.
ec3kshd3.inp	F2.
ec3kshd4.inp	F3.
ec3kshd5.inp	F4.
ec3kshd6.inp	F5.
ec3kshd7.inp	F6.
ec3kshda.inp	CORIO.
ec3kshdi.inp	HP, P, **TEMPERATURE*.
ec3kshdr.inp	ROTA.
C3D20R element load tests:	
ec3ksrd1.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, P1, P2, P3, P4, P5, P6, HP1, HP2, HP3, HP4, HP5, HP6, **TEMPERATURE*.
ec3ksrd2.inp	F1.
ec3ksrd3.inp	F2.
ec3ksrd4.inp	F3.
ec3ksrd5.inp	F4.
ec3ksrd6.inp	F5.
ec3ksrd7.inp	F6.
ec3ksrda.inp	CORIO.
ec3ksrdi.inp	HP, P, **TEMPERATURE*.
ec3ksrdr.inp	ROTA.
C3D20RH element load tests:	
ec3ksyd1.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, P1, P2, P3, P4, P5, P6, HP1, HP2, HP3, HP4, HP5, HP6, **TEMPERATURE*.
ec3ksyd2.inp	F1.
ec3ksyd3.inp	F2.
ec3ksyd4.inp	F3.
ec3ksyd5.inp	F4.
ec3ksyd6.inp	F5.
ec3ksyd7.inp	F6.
ec3ksyda.inp	CORIO.
ec3ksydi.inp	HP, P, **TEMPERATURE*.
ec3ksydr.inp	ROTA.

Three-dimensional variable number of node solids

Problem description

Model:

Cubic dimensions	$7 \times 7 \times 7$
Centrifugal and Coriolis axes of rotation	(0, 1, 0) through (−1000, 3.5, −3.5)
Gravitational load vector	(1, 0, 0)

Material:

Young's modulus	3×10^6
Poisson's ratio	0.3
Coefficient of thermal expansion	.0001
Density	10.0

Initial conditions:

Initial temperature	ALL, −10.0
Initial velocity	ALL, 1, 10.0
(Coriolis loading)	ALL, 2, 5.0
Hydrostatic pressure datum	0.0
Hydrostatic pressure elevation	−7.0

Results and discussion

The calculated reactions are in agreement with the applied loads.

Input files

C3D15V element load tests:

ec3isfd1.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, P1, P2, P3, P4, P5, HP1, HP2, HP3, HP4, HP5, **TEMPERATURE*.
ec3isfd2.inp	F1.
ec3isfd3.inp	F2.
ec3isfd4.inp	F3.
ec3isfd5.inp	F4.
ec3isfd6.inp	F5.
ec3isfda.inp	CORIO.
ec3isfdi.inp	HP, P, **TEMPERATURE*.
ec3isfdr.inp	ROTA.

C3D15VH element load tests:

ec3ishd1.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, P1, P2, P3, P4, P5, HP1, HP2, HP3, HP4, HP5, **TEMPERATURE*.

ec3ishd2.inp	F1.
ec3ishd3.inp	F2.
ec3ishd4.inp	F3.
ec3ishd5.inp	F4.
ec3ishd6.inp	F5.
ec3ishda.inp	CORIO.
ec3ishdi.inp	HP, P, **TEMPERATURE*.
ec3ishdr.inp	ROTA.

C3D27 element load tests:

ec3rsfd1.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, P1, P2, P3, P4, P5, P6, HP1, HP2, HP3, HP4, HP5, HP6, **TEMPERATURE*.
ec3rsfd2.inp	F1.
ec3rsfd3.inp	F2.
ec3rsfd4.inp	F3.
ec3rsfd5.inp	F4.
ec3rsfd6.inp	F5.
ec3rsfd7.inp	F6.
ec3rsfda.inp	CORIO.
ec3rsfdc.inp	CORIO.
ec3rsfdi.inp	HP, P, **TEMPERATURE*.
ec3rsfdr.inp	ROTA.

C3D27H element load tests:

ec3rshd1.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, P1, P2, P3, P4, P5, P6, HP1, HP2, HP3, HP4, HP5, HP6, **TEMPERATURE*.
ec3rshd2.inp	F1.
ec3rshd3.inp	F2.
ec3rshd4.inp	F3.
ec3rshd5.inp	F4.
ec3rshd6.inp	F5.
ec3rshd7.inp	F6.
ec3rshda.inp	CORIO.
ec3rshdi.inp	HP, P, **TEMPERATURE*.
ec3rshdr.inp	ROTA.

C3D27R element load tests:

ec3rshd1.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, P1, P2, P3, P4, P5, P6, HP1, HP2, HP3, HP4, HP5, HP6, **TEMPERATURE*.
ec3rshd2.inp	F1.
ec3rshd3.inp	F2.
ec3rshd4.inp	F3.
ec3rshd5.inp	F4.
ec3rshd6.inp	F5.
ec3rshd7.inp	F6.

ec3rshda.inp	CORIO.
ec3rshdi.inp	HP, P, **TEMPERATURE*.
ec3rshdr.inp	ROTA.

C3D27RH element load tests:

ec3rsyd1.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, P1, P2, P3, P4, P5, P6, HP1, HP2, HP3, HP4, HP5, HP6, **TEMPERATURE*.
ec3rsyd2.inp	F1.
ec3rsyd3.inp	F2.
ec3rsyd4.inp	F3.
ec3rsyd5.inp	F4.
ec3rsyd6.inp	F5.
ec3rsyd7.inp	F6.
ec3rsyda.inp	CORIO.
ec3rsydi.inp	HP, P, **TEMPERATURE*.
ec3rsydr.inp	ROTA.

Axisymmetric solid elements with twist

Problem description

Model:

Planar dimensions	3×3
Centrifugal axis of rotation	(0, 1, 0) through origin
Gravitational load vector	(0, −1, 0)

Material:

Young's modulus	3×10^6
Poisson's ratio	0.3
Density	5×10^{-5}

Initial conditions:

Hydrostatic pressure datum	3.0
Hydrostatic pressure elevation	0.0

Results and discussion

The calculated reactions are in agreement with the applied loads.

Input files

CGAX3 element load tests:

eca3gfd1.inp	BR, BZ, CENT, CENTRIF, GRAV, HP1, HP2, HP3, P1, P2, P3.
eca3gfdi.inp	HP, P.

CGAX3H element load tests:

eca3ghd1.inp BR, BZ, CENT, CENTRIF, GRAV, HP1, HP2, HP3, P1, P2, P3.

eca3ghdi.inp HP, P.

CGAX4 element load tests:

eca4gfd1.inp BR, BZ, CENT, CENTRIF, GRAV, HP1, HP2, HP3, HP4, P1, P2, P3, P4.

eca4gfdi.inp HP, P.

CGAX4H element load tests:

eca4ghd1.inp BR, BZ, CENT, CENTRIF, GRAV, HP1, HP2, HP3, HP4, P1, P2, P3, P4.

eca4ghdi.inp HP, P.

CGAX4R element load tests:

eca4grd1.inp BR, BZ, CENT, CENTRIF, GRAV, HP1, HP2, HP3, HP4, P1, P2, P3, P4.

eca4grdi.inp HP, P.

CGAX4RH element load tests:

eca4gyd1.inp BR, BZ, CENT, CENTRIF, GRAV, HP1, HP2, HP3, HP4, P1, P2, P3, P4.

eca4gydi.inp HP, P.

CGAX6 element load tests:

eca6gfd1.inp BR, BZ, CENT, CENTRIF, GRAV, P1, P2, P3, HP1, HP2, HP3.

eca6gfdi.inp HP, P.

CGAX6H element load tests:

eca6ghd1.inp BR, BZ, CENT, CENTRIF, GRAV, P1, P2, P3, HP1, HP2, HP3.

eca6ghdi.inp HP, P.

CGAX6M element load tests:

eca6gkd1.inp BR, BZ, CENT, CENTRIF, GRAV, P1, P2, P3, HP1, HP2, HP3.

eca6gkdi.inp HP, P.

CGAX6MH element load tests:

eca6gld1.inp BR, BZ, CENT, CENTRIF, GRAV, P1, P2, P3, HP1, HP2, HP3.

eca6gldi.inp HP, P.

CGAX8 element load tests:

eca8gfd1.inp BR, BZ, CENT, CENTRIF, GRAV, P1, P2, P3, P4, HP1, HP2, HP3, HP4.

eca8gfdi.inp HP, P.

CGAX8H element load tests:

eca8ghd1.inp BR, BZ, CENT, CENTRIF, GRAV, P1, P2, P3, P4, HP1, HP2, HP3, HP4.

eca8ghdi.inp HP, P.

CGAX8R element load tests:

eca8grd1.inp	BR, BZ, CENT, CENTRIF, GRAV, P1, P2, P3, P4, HP1, HP2, HP3, HP4.
eca8grdi.inp	HP, P.

CGAX8RH element load tests:

eca8gyd1.inp	BR, BZ, CENT, CENTRIF, GRAV, P1, P2, P3, P4, HP1, HP2, HP3, HP4.
eca8gydi.inp	HP, P.

Axisymmetric solid elements with nonlinear asymmetric deformation

Problem description

Model: Circular cross-section pipe with the global z-axis as the pipe axis.

Length	1.0
Outer radius	1.0
Wall thickness	0.5

Material:

Young's modulus	30×10^6
Poisson's ratio	0.3
Coefficient of thermal expansion	.00001

Initial conditions:

Hydrostatic pressure datum	1E6
Hydrostatic pressure elevation	0

Results and discussion

The calculated reactions are in agreement with the applied loads.

Input files

All the elements are tested with the following loads: BZ, HP1, HP2, HP3, HP4, P1, P2, P3, and P4.

Axisymmetric element tests

ecnssfd1.inp	CAXA41 element.
ecnsshd1.inp	CAXA4H1 element.
ecnssrd1.inp	CAXA4R1 element.
ecnssyd1.inp	CAXA4RH1 element.
ecntsfd1.inp	CAXA42 element.
ecntshd1.inp	CAXA4H2 element.
ecntsrd1.inp	CAXA4R2 element.

ecntsyd1.inp	CAXA4RH2 element.
ecnusfd1.inp	CAXA43 element.
ecnushd1.inp	CAXA4H3 element.
ecnusrd1.inp	CAXA4R3 element.
ecnusyd1.inp	CAXA4RH3 element.
ecnvsfd1.inp	CAXA44 element.
ecnvshd1.inp	CAXA4H4 element.
ecnvsrd1.inp	CAXA4R4 element.
ecnvsyd1.inp	CAXA4RH4 element.
ecnwsfd1.inp	CAXA81 element.
ecnwshd1.inp	CAXA8H1 element.
ecnwsrd1.inp	CAXA8R1 element.
ecnwsyd1.inp	CAXA8RH1 element.
ecnxsfd1.inp	CAXA82 element.
ecnxshd1.inp	CAXA8H2 element.
ecnxsrd1.inp	CAXA8R2 element.
ecnxsyd1.inp	CAXA8RH2 element.
ecnysfd1.inp	CAXA83 element.
ecnyshd1.inp	CAXA8H3 element.
ecnysrd1.inp	CAXA8R3 element.
ecnysyd1.inp	CAXA8RH3 element.
ecnzsfd1.inp	CAXA84 element.
ecnzshd1.inp	CAXA8H4 element.
ecnzsrd1.inp	CAXA8R4 element.
ecnzsyd1.inp	CAXA8RH4 element.

Cylindrical solid elements

Problem description

Model:

Planar dimensions	3×3
Inner radius	1
Circumferential extent	180°
Centrifugal and Coriolis axes of rotation	(0, 0, 1) through origin
Gravitational load vector	(0, 0, 1)

Material:

Young's modulus	3×10^6
Poisson's ratio	0.3
Coefficient of thermal expansion	.0001
Density	10.0

Initial conditions:

Initial temperature	ALL, –10.0
Initial velocity (Coriolis loading)	ALL, 1, 5.0
Hydrostatic pressure datum	3.0
Hydrostatic pressure elevation	0.0

Results and discussion

The calculated reactions are in agreement with the applied loads.

Input files

CCL9 element load tests:

ecc9gfd1.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, P1, P2, P3, P4, P5, HP1, HP2, HP3, HP4, HP5, **TEMPERATURE*.
ecc9gfd2.inp	F1, F2, F3, F4, F5.
ecc9gfda.inp	CORIO, ROTA.
ecc9gfdi.inp	HP, P.

CCL9H element load tests:

ecc9ghd1.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, P1, P2, P3, P4, P5, HP1, HP2, HP3, HP4, HP5, **TEMPERATURE*.
ecc9ghd2.inp	F1, F2, F3, F4, F5.
ecc9ghda.inp	CORIO, ROTA.
ecc9ghdi.inp	HP, P.

CCL12 element load tests:

eccigfd1.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, P1, P2, P3, P4, P5, P6, HP1, HP2, HP3, HP4, HP5, HP6, **TEMPERATURE*.
ecccgfd2.inp	F1, F2, F3, F4, F5, F6.
ecccgfda.inp	CORIO, ROTA.
ecccgfdi.inp	HP, P.

CCL12H element load tests:

eccighd1.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, P1, P2, P3, P4, P5, P6, HP1, HP2, HP3, HP4, HP5, HP6, **TEMPERATURE*.
ecccghd2.inp	F1, F2, F3, F4, F5, F6.
ecccghda.inp	CORIO, ROTA.
ecccghdi.inp	HP, P.

CCL18 element load tests:

eccigfd1.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, P1, P2, P3, P4, P5, HP1, HP2, HP3, HP4, HP5, **TEMPERATURE*.
eccigfd2.inp	F1, F2, F3, F4, F5.
eccigfda.inp	CORIO, ROTA.
eccigfdi.inp	HP, P.

CCL18H element load tests:

eccighd1.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, P1, P2, P3, P4, P5, HP1, HP2, HP3, HP4, HP5, **TEMPERATURE*.
eccighd2.inp	F1, F2, F3, F4, F5.
eccighda.inp	CORIO, ROTA.
eccighdi.inp	HP, P.

CCL24 element load tests:

eccrgfd1.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, P1, P2, P3, P4, P5, P6, HP1, HP2, HP3, HP4, HP5, HP6, **TEMPERATURE*.
eccrgfd2.inp	F1, F2, F3, F4, F5, F6.
eccrgfda.inp	CORIO, ROTA.
eccrgfdi.inp	HP, P.

CCL24H element load tests:

eccrghd1.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, P1, P2, P3, P4, P5, P6, HP1, HP2, HP3, HP4, HP5, HP6, **TEMPERATURE*.
eccrghd2.inp	F1, F2, F3, F4, F5, F6.
eccrghda.inp	CORIO, ROTA.
eccrghdi.inp	HP, P.

CCL24R element load tests:

eccrgrd1.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, P1, P2, P3, P4, P5, P6, HP1, HP2, HP3, HP4, HP5, HP6, **TEMPERATURE*.
eccrgrd2.inp	F1, F2, F3, F4, F5, F6.
eccrgrda.inp	CORIO, ROTA.
eccrgrdi.inp	HP, P.

CCL24RH element load tests:

eccrgyd1.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, P1, P2, P3, P4, P5, P6, HP1, HP2, HP3, HP4, HP5, HP6, **TEMPERATURE*.
eccrgyd2.inp	F1, F2, F3, F4, F5, F6.
eccrgyda.inp	CORIO, ROTA.
eccrgydi.inp	HP, P.

Field expansion tests

Problem description

Model: This section lists a number of simple tests that verify the field expansion capability. In most cases a single element or a small assembly of elements is loaded using the field expansion capability.

Material: Most tests use a linear elastic material model. There are a few tests that use a hyperelastic material model. In all cases a field expansion coefficient is defined and is associated with at least one, and in some cases more than one, predefined field variable.

Initial conditions: In all cases the initial value of all relevant field variables is assumed to be zero at all the nodes.

Results and discussion

The results for loading based on field expansion match those obtained from a similar model using thermal expansion.

Input files

fieldexp_cpe4.inp	CPE4 element using a linear elastic material model and loaded with both field and thermal expansion.
fieldexp_cps4.inp	CPS4 element using a linear elastic material and loaded with both field and thermal expansion.
fieldexp_c3d8.inp	C3D8 element using a linear elastic material and loaded with both field and thermal expansion.
hyper-field-expand.inp	C3D8 element using a hyperelastic material loaded with field expansion driven by a single field variable. Tests nonlinear static, static perturbation, and buckle procedures.
hyper-thermfield-expand.inp	C3D8S element using a hyperelastic material loaded with both field and thermal expansion. The field expansion is driven by a single field variable. Tests nonlinear static, static perturbation, and buckle procedures.
hyper-twofield-expand.inp	C3D8 element using a hyperelastic material loaded with field expansion driven by two different field variables. Tests nonlinear static, static perturbation, and buckle procedures.

Beam stress/displacement elements

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Load types: CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, predefined temperature, ROTA

Problem description

Model:

Length	15.0
Centrifugal axis of rotation	(0, 0, 1) through (7.5, 0, 0)
Gravity load vector	(1, 0, 0)
Beam section data:	
Arbitrary (closed)	n = 4, A = (–.995, 1.49), B = (–.995, –1.49), t_{AB} = 0.01, C = (.995, –1.49), t_{BC} = 0.02, D = (.995, 1.49), t_{CD} = 0.01, E = (–.995, 1.49), t_{DE} = 0.02
Arbitrary (open)	n = 2, A = (0.0, 3.95), B = (0.0, 0.0), t_{AB} = 0.1, C = (3.95, 0.0), t_{BC} = 0.1
Box	a = 2.0, b = 3.0, $t_1 = t_3$ = 0.01, $t_2 = t_4$ = 0.02
Circle	r = 2.0
General	A = 12.566, $I_{11} = I_{22}$ = 12.566, J = 25.133
Hexagonal	r = 2, t = 0.02
I-section	h = 2.4, l = 1.2, b_1 = 3.0, b_2 = 2.0, $t_1 = t_2 = t_3$ = 0.02
L-section	a = 4.0, b = 4.0, c = 0.1, d = 0.1
Pipe	r = 2.0, t = 0.2
Rectangular	a = 2.0, b = 3.0
Trapezoidal	a = 2.0, b = 3.0, c = 2.0, d = 1.5

Material:

Young's modulus	3×10^6
Poisson's ratio	0.3
Density	0.16667
Coefficient of thermal expansion	0.0001

Initial conditions:

Initial temperature	ALL, −10.0

Results and discussion

The calculated reactions are in agreement with the applied loads.

Input files

Rectangular section

B21 element load tests:

eb22qxd1.inp	CENT, CENTRIF, GRAV, PX, PY, P2, *TEMPERATURE.
eb22qxdi.inp	P, *TEMPERATURE.
eb22rgd1.inp	CENT, CENTRIF, GRAV, PX, PY, P2, *TEMPERATURE.
eb22rgdi.inp	P, *TEMPERATURE.
eb22rvd1.inp	CENT, CENTRIF, GRAV, PX, PY, P2, *TEMPERATURE.
eb22rvdi.inp	P, *TEMPERATURE.
eb22rxdr.inp	ROTA.

B21H element load tests:

eb2hqxd1.inp	CENT, CENTRIF, GRAV, PX, PY, P2, *TEMPERATURE.
eb2hqxdi.inp	P, *TEMPERATURE.
eb2hrgd1.inp	CENT, CENTRIF, GRAV, PX, PY, P2, *TEMPERATURE.
eb2hrgdi.inp	P, *TEMPERATURE.
eb2hrvd1.inp	CENT, CENTRIF, GRAV, PX, PY, P2, *TEMPERATURE.
eb2hrvdi.inp	P, *TEMPERATURE.
eb2hrxdr.inp	ROTA.

B22 element load tests:

eb23qxd1.inp	CENT, CENTRIF, GRAV, PX, PY, P2,*TEMPERATURE.
eb23qxdi.inp	P, *TEMPERATURE.
eb23rgd1.inp	CENT, CENTRIF, GRAV, PX, PY, P2, *TEMPERATURE.
eb23rgdi.inp	P, *TEMPERATURE.
eb23rvd1.inp	CENT, CENTRIF, GRAV, PX, PY, P2, *TEMPERATURE.
eb23rvdi.inp	P, *TEMPERATURE.
eb23rxdr.inp	ROTA.

B22H element load tests:

eb2iqxd1.inp	CENT, CENTRIF, GRAV, PX, PY, P2, *TEMPERATURE.
eb2iqxdi.inp	P, *TEMPERATURE.
eb2irgd1.inp	CENT, CENTRIF, GRAV, PX, PY, P2, *TEMPERATURE.
eb2irgdi.inp	P, *TEMPERATURE.
eb2irvd1.inp	CENT, CENTRIF, GRAV, PX, PY, P2, *TEMPERATURE.
eb2irvdi.inp	P, *TEMPERATURE.

eb2irxdr.inp	ROTA.

B23 element load tests:

eb2aqxd1.inp	CENT, CENTRIF, GRAV, PX, PY, P2, *TEMPERATURE.
eb2aqxdi.inp	P, *TEMPERATURE.
eb2argd1.inp	CENT, CENTRIF, GRAV, PX, PY, P2, *TEMPERATURE.
eb2argdi.inp	P, *TEMPERATURE.
eb2arvd1.inp	CENT, CENTRIF, GRAV, PX, PY, P2, *TEMPERATURE.
eb2arvdi.inp	P, *TEMPERATURE.
eb2arxdr.inp	ROTA.

B23H element load tests:

eb2jqxd1.inp	CENT, CENTRIF, GRAV, PX, PY, P2, *TEMPERATURE.
eb2jqxdi.inp	P, *TEMPERATURE.
eb2jrgd1.inp	CENT, CENTRIF, GRAV, PX, PY, P2, *TEMPERATURE.
eb2jrgdi.inp	P, *TEMPERATURE.
eb2jrvd1.inp	CENT, CENTRIF, GRAV, PX, PY, P2, *TEMPERATURE.
eb2jrvdi.inp	P, *TEMPERATURE.
eb2jrxdr.inp	ROTA.

B31 element load tests:

eb32qxd1.inp	CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.
eb32rgd1.inp	CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.
eb32rvd1.inp	CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.
eb32rxdr.inp	ROTA.

B31H element load tests:

eb3hqxd1.inp	CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.
eb3hrgd1.inp	CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.
eb3hrvd1.inp	CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.
eb3hrxdr.inp	ROTA.

B32 element load tests:

eb33qxd1.inp	CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.
eb33rgd1.inp	CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.
eb33rvd1.inp	CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.
eb33rxdr.inp	ROTA.

B32H element load tests:

eb3iqxd1.inp	CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.
eb3irgd1.inp	CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.
eb3irvd1.inp	CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.
eb3irxdr.inp	ROTA.

B33 element load tests:

eb3aqxd1.inp	CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.
eb3argd1.inp	CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.

eb3arvd1.inp CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.
eb3arxdr.inp ROTA.

B33H element load tests:

eb3jqxd1.inp CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.
eb3jrgd1.inp CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.
eb3jrvd1.inp CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.
eb32rxdr.inp ROTA.

I-section

B22H element load tests:

eb2iigd1.inp CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.
eb2iigdi.inp P, *TEMPERATURE.
eb2iivd1.inp CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.
eb2iivdi.inp P, *TEMPERATURE.
eb2ikxd1.inp CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.
eb2ikxdi.inp P, *TEMPERATURE.

B31OS element load tests:

ebo2igd1.inp CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.
ebo2ivd1.inp CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.
ebo2ixdr.inp ROTA.
ebo2kxd1.inp CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.

B31OSH element load tests:

ebohigd1.inp CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.
ebohivd1.inp CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.
ebohixdr.inp ROTA.
ebohkxd1.inp CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.

B32H element load tests:

eb3iigd1.inp CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.
eb3iivd1.inp CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.
eb3ikxd1.inp CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.

B32OS element load tests:

ebo3igd1.inp CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.
ebo3ivd1.inp CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.
ebo3ixdr.inp ROTA.
ebo3kxd1.inp CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.

B32OSH element load tests:

eboiigd1.inp CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.
eboiivd1.inp CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.
eboiixdr.inp ROTA.
eboikxd1.inp CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.

Box section, arbitrary closed section

B22H element load tests:

eb2ibgd1.inp	CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.
eb2ibgdi.inp	P, *TEMPERATURE.
eb2ibvd1.inp	CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.
eb2ibvdi.inp	P, *TEMPERATURE.
eb2iexd1.inp	CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.
eb2iexdi.inp	P, *TEMPERATURE.

B32H element load tests:

eb3ibgd1.inp	CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.
eb3ibvd1.inp	CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.
eb3iexd1.inp	CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.
eb3iabd1.inp	CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.
eb3ia1d1.inp	CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.
eb3idbd1.inp	CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.

Circular section

B22H element load tests:

eb2icgd1.inp	CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.
eb2icgdi.inp	P, *TEMPERATURE.
eb2icvd1.inp	CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.
eb2icvdi.inp	P, *TEMPERATURE.
eb2ifxd1.inp	CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.
eb2ifxdi.inp	P, *TEMPERATURE.

B32H element load tests:

eb3icgd1.inp	CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.
eb3icvd1.inp	CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.
eb3ifxd1.inp	CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.

General section

B22H element load tests:

eb2igxd1.inp	CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.
eb2igxdi.inp	P, *TEMPERATURE.

B32H element load test:

eb3igxd1.inp	CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.

Hexagonal section

B22H element load tests:

eb2ihgd1.inp	CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.
eb2ihgdi.inp	P, *TEMPERATURE.
eb2ihvd1.inp	CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.
eb2ijxd1.inp	CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.

eb2ijxdi.inp P, *TEMPERATURE.

B32H element load tests:

eb3ihgd1.inp CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.
eb3ihvd1.inp CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.
eb3ijxd1.inp CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.

L-section, arbitrary open section

B32H element load tests:

eb3ilgd1.inp CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.
eb3ilvd1.inp CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.
eb3imxd1.inp CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.
eb3iald1.inp CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.
eb3ia2d1.inp CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.
eb3idld1.inp CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.

Pipe section

B22H element load tests:

eb2ipgd1.inp CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.
eb2ipgdi.inp P, *TEMPERATURE.
eb2ipvd1.inp CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.
eb2ipvdi.inp P, *TEMPERATURE.
eb2ioxd1.inp CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.
eb2ioxdi.inp P, *TEMPERATURE.

B32H element load tests:

eb3ipgd1.inp CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.
eb3ipvd1.inp CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.
eb3ioxd1.inp CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.

Trapezoidal section

B22H element load tests:

eb2itgd1.inp CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.
eb2itgdi.inp P, *TEMPERATURE.
eb2itvd1.inp CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.
eb2itvdi.inp P, *TEMPERATURE.
eb2isxd1.inp CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.
eb2isxdi.inp P, *TEMPERATURE.

B32H element load tests:

eb3itgd1.inp CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.
eb3itvd1.inp CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.
eb3isxd1.inp CENT, CENTRIF, GRAV, PX, PY, PZ, P1, P2, *TEMPERATURE.

Load types: F1, F2

Problem description

Model:

Length	15.0
Rectangular section data	$a = 2.0$, $b = 3.0$
I-section data	$h = 2.4$, $l = 1.2$, $b_1 = 3.0$, $b_2 = 2.0$, $t_1 = t_2 = t_3 = 0.2$

Material:

Young's modulus	3×10^6
Poisson's ratio	0.3

Results and discussion

The calculated reactions are in agreement with the applied loads.

Input files

Rectangular section

eb22rxd3.inp	B21: F2.
eb2hrxd3.inp	B21H: F2.
eb23rxd3.inp	B22: F2.
eb2irxd3.inp	B22H: F2.
eb2arxd3.inp	B23: F2.
eb2jrxd3.inp	B23H: F2.
eb32rxd2.inp	B31: F1.
eb32rxd3.inp	B31: F2.
eb3hrxd2.inp	B31H: F1.
eb3hrxd3.inp	B31H: F2.
eb33rxd2.inp	B32: F1.
eb33rxd3.inp	B32: F2.
eb3irxd2.inp	B32H: F1.
eb3irxd3.inp	B32H: F2.
eb3arxd2.inp	B33: F1.
eb3arxd3.inp	B33: F2.
eb3jrxd2.inp	B33H: F1.
eb3jrxd3.inp	B33H: F2.

I-section

ebo2ixd2.inp	B31OS: F1.

ebo2ixd3.inp	B31OS: F2.
ebohixd2.inp	B31OSH: F1.
ebohixd3.inp	B31OSH: F2.
ebo3ixd2.inp	B32OS: F1.
ebo3ixd3.inp	B32OS: F2.
eboiixd2.inp	B32OSH: F1.
eboiixd3.inp	B32OSH: F2.

Foundation types: FX, FY, FZ

Problem description

Model:

Length	$10\sqrt{2}$
Orientation	45° with horizontal axis
Pipe section data	$r = 1.0$, $t = 0.05$
I-section data	$h = 2.4$, $l = 1.2$, $b_1 = 3.0$, $b_2 = 2.0$, $t_1 = t_2 = t_3 = 0.2$

Material:

Young's modulus	30×10^6
Poisson's ratio	0.3

Results and discussion

The calculated reactions are in agreement with the applied loads.

Input files

Pipe section

eb22pxd9.inp	B21: FX, FY.
eb2hpxd9.inp	B21H: FX, FY.
eb23pxd9.inp	B22: FX, FY.
eb2ipxd9.inp	B22H: FX, FY.
eb2apxd9.inp	B23: FX, FY.
eb2jpxd9.inp	B23H: FX, FY.
eb32pxd9.inp	B31: FX, FY, FZ.
eb3hpxd9.inp	B31H: FX, FY, FZ.
eb33pxd9.inp	B32: FX, FY, FZ.
eb3ipxd9.inp	B32H: FX, FY, FZ.
eb3apxd9.inp	B33: FX, FY, FZ.
eb3jpxd9.inp	B33H: FX, FY, FZ.

I-section

ebo2ixd9.inp	B31OS: FX, FY, FZ.
ebohixd9.inp	B31OSH: FX, FY, FZ.
ebo3ixd9.inp	B32OS: FX, FY, FZ.
eboiixd9.inp	B32OSH: FX, FY, FZ.

Coriolis loading

Problem description

Model:

Pipe section data	$r = 10.0$, $t = 1.0$
I-section data	$h = 2.4$, $l = 1.2$, $b_1 = 3.0$, $b_2 = 2.0$, $t_1 = t_2 = t_3 = 0.2$
Axis of rotation	(0, 0, 1) through (0, 0, 0)

Material:

Young's modulus	30×10^6

Initial conditions:

Initial velocity	ALL, 1, 10.0
	ALL, 2, 5.0
	ALL, 3, 2.0 (for 3D beams)

Results and discussion

The calculated reactions are in agreement with the applied loads.

Input files

All elements are tested with the CORIO load.

Pipe section

eb22pxda.inp	B21 element.
eb2hpxda.inp	B21H element.
eb23pxda.inp	B22 element.
eb2ipxda.inp	B22H element.
eb2apxda.inp	B23 element.
eb2jpxda.inp	B23H element.
eb32pxda.inp	B31 element.
eb3hpxda.inp	B31H element.
eb33pxda.inp	B32 element.
eb3ipxda.inp	B32H element.
eb3apxda.inp	B33 element.

eb3jpxda.inp	B33H element.

I-section

ebo2ixda.inp	B31OS element.
ebohixda.inp	B31OSH element.
ebo3ixda.inp	B32OS element.
eboiixda.inp	B32OSH element.

Pipe stress/displacement elements

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Distributed loads

Problem description

Internal pressures are applied to an effective diameter of 3.6, whereas external pressures are applied to an effective diameter of 4.0. The effective axial force output variable ESF1 is also tested.

Model:

Length	15.0
Pipe section data	$r = 2.0$, $t = 0.2$
Centrifugal axis of rotation	(0, 0, 1) through (7.5, 0, 0)
Gravity load vector	(0, 1, 0)

Material:

Young's modulus	3×10^6
Poisson's ratio	0.3
Density	0.4188

Initial conditions:

Initial temperature	ALL, −10.0
Initial velocity	ALL, 1, 10.0
(Coriolis loading)	ALL, 2, 5.0
Hydrostatic pressure datum	100.0
Hydrostatic pressure elevation	0.0

Results and discussion

The calculated reactions are in agreement with the applied loads.

Input files

PIPE21 element load tests:

ep22pxd1.inp	CENT, CENTRIF, P2, PI, PE, HPI, HPE, PX, PY, GRAV, CORIO, **TEMPERATURE*.
ep22pxd3.inp	F2.

ep22pxdr.inp	ROTA.

PIPE21H element load tests:

ep2hpxd1.inp	CENT, CENTRIF, P2, PI, PE, HPI, HPE, PX, PY, GRAV, CORIO, **TEMPERATURE*.
ep2hpxd3.inp	F2.
ep2hpxdr.inp	ROTA.

PIPE22 element load tests:

ep23pxd1.inp	CENT, CENTRIF, P2, PI, PE, HPI, HPE, PX, PY, GRAV, CORIO, **TEMPERATURE*.
ep23pxd3.inp	F2.
ep23pxdr.inp	ROTA.

PIPE22H element load tests:

ep2ipxd1.inp	CENT, CENTRIF, P2, PI, PE, HPI, HPE, PX, PY, GRAV, CORIO, **TEMPERATURE*.
ep2ipxd3.inp	F2.
ep2ipxdr.inp	ROTA.

PIPE31 element load tests:

ep32pxd1.inp	CENT, CENTRIF, P1, P2, PI, PE, HPI, HPE, PX, PY, PZ, GRAV, CORIO, **TEMPERATURE*.
ep32pxd2.inp	F1.
ep32pxd3.inp	F2, perturbation step with **LOAD CASE*.
ep32pxdr.inp	ROTA.

PIPE31H element load tests:

ep3hpxd1.inp	CENT, CENTRIF, P1, P2, PI, PE, HPI, HPE, PX, PY, PZ, GRAV, CORIO, **TEMPERATURE*.
ep3hpxd2.inp	F1.
ep3hpxd3.inp	F2.
ep3hpxdr.inp	ROTA.

PIPE32 element load tests:

ep33pxd1.inp	CENT, CENTRIF, P1, P2, PI, PE, HPI, HPE, PX, PY, PZ, GRAV, CORIO, **TEMPERATURE*.
ep33pxd2.inp	F1.
ep33pxd3.inp	F2.
ep33pxdr.inp	ROTA.

PIPE32H element load tests:

ep3ipxd1.inp	CENT, CENTRIF, P1, P2, PI, PE, HPI, HPE, PX, PY, PZ, GRAV, CORIO, **TEMPERATURE*.
ep3ipxd2.inp	F1.
ep3ipxd3.inp	F2.
ep3ipxdr.inp	ROTA.

PIPE21 element load tests in Abaqus/Explicit:

pipe21_load_options_xpl.inp P2, PI, PE, HPI, HPE, PX, PY, GRAV, **TEMPERATURE*.

PIPE31 element load tests in Abaqus/Explicit:

pipe31_load_options_xpl.inp P1, P2, PI, PE, HPI, HPE, PX, PY, PZ, GRAV, **TEMPERATURE*.

Foundation types: FX, FY, FZ

Problem description

Model:

Length	$10\sqrt{2}$
Orientation	45° with horizontal axis
Pipe section data	$r = 1.0$, $t = 0.05$

Material:

Young's modulus	30×10^6
Poisson's ratio	0.3

Results and discussion

The calculated reactions are in agreement with the applied loads.

Input files

ep22pxd9.inp	PIPE21: FX, FY.
ep2hpxd9.inp	PIPE21H: FX, FY.
ep23pxd9.inp	PIPE22: FX, FY.
ep2ipxd9.inp	PIPE22H: FX, FY.
ep32pxd9.inp	PIPE31: FX, FY, FZ.
ep3hpxd9.inp	PIPE31H: FX, FY, FZ.
ep33pxd9.inp	PIPE32: FX, FY, FZ.
ep3ipxd9.inp	PIPE32H: FX, FY, FZ.

Effective axial force with modal procedures

Elements tested

PIPE21

Problem description

Internal pressures are applied to an effective area of 1.0, while external pressures are applied to an effective area of 2.0. The effective axial force output variable ESF1 is also tested.

Model:

Length	100.0

Orientation	90° and 45° with horizontal axis
Pipe section data	$r = 1.0$, $t = 0.25$

Material:

Young's modulus	1×10^6
Poisson's ratio	0.0
Density	1.0

Results and discussion

The effective axial force output, ESF1, agrees with the analytically determined values, which are documented at the top of the input file.

Input files

xesf1mod.inp	Input file for this analysis.

Shell, membrane, and truss stress/displacement elements

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Axisymmetric shells

Problem description

Model:

Length	10.0
Radius	5.0
Thickness	0.5
Centrifugal axis of rotation	(0, 1, 0) through origin
Gravity load vector	(0, 1, 0)

Material:

Young's modulus	3×10^6
Poisson's ratio	0.3
Density	1.0

Initial conditions:

Hydrostatic pressure datum	12.0
Hydrostatic pressure elevation	0.0

General: Gauss integration is used for the shell cross-section in input file *esa2sxd1.inp*.

Results and discussion

The calculated reactions are in agreement with the applied loads.

Input files

SAX1 element load tests:

esa2sxd1.inp	BR, BZ, GRAV, CENT, CENTRIF, P, HP.
esa2sxd8.inp	F.
esa2sxdi.inp	P, HP.

SAX2 element load tests:

esa3sxd1.inp	BR, BZ, GRAV, CENT, CENTRIF, P, HP.

esa3sxd8.inp	F.
esa3sxdi.inp	P, HP.

Axisymmetric membranes

Problem description

Model:

Length	10.0
Radius	5.0
Thickness	0.5
Centrifugal axis of rotation	(0, 1, 0) through origin
Gravity load vector	(0, 1, 0)

Material:

Young's modulus	3×10^6
Poisson's ratio	0.3
Density	1.0

Initial conditions:

Hydrostatic pressure datum	12.0
Hydrostatic pressure elevation	0.0

Results and discussion

The calculated reactions are in agreement with the applied loads.

Input files

MAX1 element load tests:

ema2srd1.inp	BR, BZ, GRAV, CENT, CENTRIF, P, HP.
ema2srd8.inp	F.
ema2srdi.inp	P, HP.

MAX2 element load tests:

ema3srd1.inp	BR, BZ, GRAV, CENT, CENTRIF, P, HP.
ema3srd8.inp	F.
ema3srdi.inp	P, HP.

MGAX1 element load tests:

emg2srd1.inp	BR, BZ, GRAV, CENT, CENTRIF, P, HP.
emg2srd8.inp	F.
emg2srdi.inp	P, HP.

MGAX2 element load tests:

emg3srd1.inp	BR, BZ, GRAV, CENT, CENTRIF, P, HP.
emg3srd8.inp	F.
emg3srdi.inp	P, HP.

CYLINDRICAL membranes

Problem description

Model:

Length	10.0
Radius	5.0
Thickness	0.5
Centrifugal axis of rotation	(0, 0, 1) through origin
Coriolis axis of rotation	(0, 0, 1) through origin
Gravity load vector	(0, 0, 1)

Material:

Young's modulus	3×10^6
Poisson's ratio	0.3
Density	1.0

Initial conditions:

Hydrostatic pressure datum	12.0
Hydrostatic pressure elevation	0.0

Results and discussion

The calculated reactions are in agreement with the applied loads.

Input files

MCL6 element load tests:

emc6srd1.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, P, HP.
emc6srd8.inp	F.
emc6srda.inp	CORIO.
emc6srdr.inp	ROTA.

MCL9 element load tests:

emc9srd1.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, P, HP.
emc9srd8.inp	F.
emc9srda.inp	CORIO.
emc9srdr.inp	ROTA.

General shells and membranes: general element loading

Problem description

Model:

Square dimensions	7×7
Thickness	2.0
Centrifugal axis of rotation	(0, 1, 0) through origin
Coriolis axis of rotation	(0, 0, 1) through origin
Gravity load vector	(0, 0, 1)

Material:

Young's modulus	3×10^6
Poisson's ratio	0.3
Density	1.0
Coefficient of thermal expansion	.0001

Initial conditions:

Initial temperature	ALL, −10
Hydrostatic pressure datum	7.0
Hydrostatic pressure elevation	0.0
Initial velocity	ALL, 1, 10.0
(Coriolis loading)	ALL, 2, 5.0

Results and discussion

The calculated reactions are in agreement with the applied loads.

Input files

S3/S3R element load tests:

esf3sgd1.inp	BX, BY, BZ, GRAV, CENT, P, HP, **TEMPERATURE*.
esf3sgdi.inp	P, HP, **TEMPERATURE*.
esf3sxd1.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, P, HP, **TEMPERATURE*.
esf3sxdi.inp	P, HP, **TEMPERATURE*.
esf3sxdr.inp	ROTA.

S4 element load tests:

ese4sgd1.inp	BX, BY, BZ, GRAV, CENT, P, HP, **TEMPERATURE*.
ese4sgdi.inp	P, HP, **TEMPERATURE*.
ese4sxd1.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, P, HP, **TEMPERATURE*.
ese4sxd8.inp	F.

ese4sxda.inp	CORIO.
ese4sxdi.inp	P, HP, **TEMPERATURE*.

S4R element load tests:

esf4sgd1.inp	BX, BY, BZ, GRAV, CENT, P, HP, **TEMPERATURE*.
esf4sgdi.inp	P, HP, **TEMPERATURE*.
esf4sxd1.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, P, HP, **TEMPERATURE*.
esf4sxd8.inp	F.
esf4sxda.inp	CORIO.
esf4sxdi.inp	P, HP, **TEMPERATURE*.
esf4sxdr.inp	ROTA.

S4R5 element load tests:

es54sgd1.inp	BX, BY, BZ, GRAV, CENT, P, HP, **TEMPERATURE*.
es54sgdi.inp	P, HP, **TEMPERATURE*.
es54sxd1.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, P, HP, **TEMPERATURE*.
es54sxd8.inp	F.
es54sxda.inp	CORIO.
es54sxdi.inp	P, HP, **TEMPERATURE*.
es54sxdr.inp	ROTA.

S8R element load tests:

es68sgd1.inp	BX, BY, BZ, GRAV, CENT, P, HP, **TEMPERATURE*.
es68sgdi.inp	P, HP, **TEMPERATURE*.
es68sxd1.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, P, HP, **TEMPERATURE*.
es68sxd8.inp	F.
es68sxda.inp	CORIO.
es68sxdi.inp	P, HP, **TEMPERATURE*.
es68sxdr.inp	ROTA.

S8R5 element load tests:

es58sgd1.inp	BX, BY, BZ, GRAV, CENT, P, HP, **TEMPERATURE*.
es58sgdi.inp	P, HP, **TEMPERATURE*.
es58sxd1.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, P, HP, **TEMPERATURE*.
es58sxd8.inp	F.
es58sxda.inp	CORIO.
es58sxdi.inp	P, HP, **TEMPERATURE*.
es58sxdr.inp	ROTA.

S9R5 element load tests:

es59sgd1.inp	BX, BY, BZ, GRAV, CENT, P, HP, **TEMPERATURE*.
es59sgdi.inp	P, HP, **TEMPERATURE*.
es59sxd1.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, P, HP, **TEMPERATURE*.
es59sxd8.inp	F.
es59sxda.inp	CORIO.
es59sxdi.inp	P, HP, **TEMPERATURE*.

es59sxdr.inp	ROTA.
STRI3 element load tests:	
es63sgd1.inp	BX, BY, BZ, GRAV, CENT, P, HP, **TEMPERATURE*.
es63sgdi.inp	P, HP, **TEMPERATURE*.
es63sxd1.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, P, HP, **TEMPERATURE*.
es63sxd8.inp	F.
es63sxda.inp	CORIO.
es63sxdi.inp	P, HP, **TEMPERATURE*.
es63sxdr.inp	ROTA.
STRI65 element load tests:	
es56sgd1.inp	BX, BY, BZ, GRAV, CENT, P, HP, **TEMPERATURE*.
es56sgdi.inp	P, HP, **TEMPERATURE*.
es56sxd1.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, P, HP, **TEMPERATURE*.
es56sxd8.inp	F.
es56sxda.inp	CORIO.
es56sxdi.inp	P, HP, **TEMPERATURE*.
es56sxdr.inp	ROTA.
M3D3 element load tests:	
em33sfd1.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, P, HP, **TEMPERATURE*.
em33sfd8.inp	F.
em33sfda.inp	CORIO.
em33sfdi.inp	P, HP, **TEMPERATURE*.
em33sfdr.inp	ROTA.
M3D4 element load tests:	
em34sfd1.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, P, HP, **TEMPERATURE*.
em34sfd8.inp	F.
em34sfda.inp	CORIO.
em34sfdi.inp	P, HP, **TEMPERATURE*.
em34sfdr.inp	ROTA.
M3D4R element load tests:	
em34srd1.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, P, HP, **TEMPERATURE*.
em34srd8.inp	F.
em34srda.inp	CORIO.
em34srdi.inp	P, HP, **TEMPERATURE*.
em34srdr.inp	ROTA.
M3D6 element load tests:	
em36sfd1.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, P, HP, **TEMPERATURE*.

em36sfd8.inp	F.
em36sfda.inp	CORIO.
em36sfdi.inp	P, HP, **TEMPERATURE*.
em36sfdr.inp	ROTA.
M3D8 element load tests:	
em38sfd1.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, P, HP, **TEMPERATURE*.
em38sfd8.inp	F.
em38sfda.inp	CORIO.
em38sfdi.inp	P, HP, **TEMPERATURE*.
em38sfdr.inp	ROTA.
M3D8R element load tests:	
em38srd1.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, P, HP, **TEMPERATURE*.
em38srd8.inp	F.
em38srda.inp	CORIO.
em38srdi.inp	P, HP, **TEMPERATURE*.
em38srdr.inp	ROTA.
M3D9 element load tests:	
em39sfd1.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, P, HP, **TEMPERATURE*.
em39sfd8.inp	F.
em39sfda.inp	CORIO.
em39sfdi.inp	P, HP, **TEMPERATURE*.
em39sfdr.inp	ROTA.
M3D9R element load tests:	
em39srd1.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, P, HP, **TEMPERATURE*.
em39srd8.inp	F.
em39srda.inp	CORIO.
em39srdi.inp	P, HP, **TEMPERATURE*.
em39srdr.inp	ROTA.

General shells and membranes: unconstrained thermal expansion

Problem description

Unconstrained expansion of a hollow cylinder subject to uniform thermal loading is investigated. One-quarter of the cylinder is modeled with a 6×6 mesh of quadrilateral elements with appropriate boundary conditions applied along lines of symmetry. A similar discretization is used (with the diagonals crossed on the quadrilaterals) to test triangular elements.

Model:

Length	0.405
Radius	0.2875
Thickness	0.05

Material:

Coefficient of thermal expansion	4.87×10^{-6}

Initial conditions:

Initial temperature	ALL, 70.0

Results and discussion

The calculated reactions are in agreement with the applied loads.

Input files

esf3sxdg.inp	S3/S3R: **TEMPERATURE*.
ese4sgdg.inp	S4: **TEMPERATURE*.
ese4sxdg.inp	S4: **TEMPERATURE*.
esf4sxdg.inp	S4R: **TEMPERATURE*.
es54sxdg.inp	S4R5: **TEMPERATURE*.
es68sxdg.inp	S8R: **TEMPERATURE*.
es58sxdg.inp	S8R5: **TEMPERATURE*.
es59sxdg.inp	S9R5: **TEMPERATURE*.
es63sxdg.inp	STRI3: **TEMPERATURE*.
es56sxdg.inp	STRI65: **TEMPERATURE*.

Axisymmetric shells with nonlinear asymmetric deformation

Problem description

Model:

Length	10.0
Radius	5.0
Thickness	0.01

Material:

Young's modulus	3×10^7
Poisson's ratio	0.3
Density	1.0

Initial conditions:

Hydrostatic pressure datum	1×10^6
Hydrostatic pressure elevation	0.0

Results and discussion

The calculated reactions are in agreement with the applied loads.

Input files

esnssxd1.inp	SAXA11: BX, BZ, HP, P.
esntsxd1.inp	SAXA12: BX, BZ, HP, P.
esnusxd1.inp	SAXA13: BX, BZ, HP, P.
esnvsxd1.inp	SAXA14: BX, BZ, HP, P.
esnwsxd1.inp	SAXA21: BX, BZ, HP, P.
esnxsxd1.inp	SAXA22: BX, BZ, HP, P.
esnysxd1.inp	SAXA23: BX, BZ, HP, P.
esnzsxd1.inp	SAXA24: BX, BZ, HP, P.

Truss elements

Problem description

Model:

Length	1.0
Area	0.1
Centrifugal axis of rotation	(0, 1, 0) through (.5, 0, 0)
Gravitational load vector	(0, −1, 0)

Material:

Young's modulus	3×10^6
Poisson's ratio	0.3
Coefficient of thermal expansion	.0001
Density	5×10^{-5}

Initial conditions:

Initial temperature	ALL, −10.0
Initial velocity	ALL, 1, 10.0
(Coriolis loading)	ALL, 2, 5.0
(3D only)	ALL, 3, 2.0

Results and discussion

The calculated reactions are in agreement with the applied loads.

Input files

T2D2 element load tests:

et22sfd1.inp	BX, BY, CENT, CENTRIF, GRAV, **TEMPERATURE*.
et22sfda.inp	CORIO.
et22sfdc.inp	CORIO.
et22sfdr.inp	ROTA.

T2D2H element load tests:

et22shd1.inp	BX, BY, CENT, CENTRIF, GRAV, **TEMPERATURE*.
et22shda.inp	CORIO.
et22shdr.inp	ROTA.

T2D3 element load tests:

et23sfd1.inp	BX, BY, CENT, CENTRIF, GRAV, **TEMPERATURE*.
et23sfda.inp	CORIO.
et23sfdc.inp	CORIO.
et23sfdr.inp	ROTA.

T2D3H element load tests:

et23shd1.inp	BX, BY, CENT, CENTRIF, GRAV, **TEMPERATURE*.
et23shda.inp	CORIO.
et23shdr.inp	ROTA.

T3D2 element load tests:

et32sfd1.inp	BX, BY, BZ, CENT, CENTRIF, GRAV, **TEMPERATURE*.
et32sfda.inp	CORIO.
et32sfdc.inp	CORIO.
et32sfdr.inp	ROTA.

T3D2H element load tests:

et32shd1.inp	BX, BY, BZ, CENT, CENTRIF, GRAV, **TEMPERATURE*.
et32shda.inp	CORIO.
et32shdr.inp	ROTA.

T3D3 element load tests:

et33sfd1.inp	BX, BY, BZ, CENT, CENTRIF, GRAV, **TEMPERATURE*.
et33sfda.inp	CORIO.
et33sfdc.inp	CORIO.
et33sfdr.inp	ROTA.

T3D3H element load tests:

et33shd1.inp	BX, BY, BZ, CENT, CENTRIF, GRAV, **TEMPERATURE*.
et33shda.inp	CORIO.

et33shdr.inp	ROTA.

Field expansion tests

Problem description

Model: This section lists a number of simple tests that verify the field expansion capability. In most cases a single element or a small assembly of elements is loaded using the field expansion capability.

Material: All tests use a linear elastic material model. In all cases a field expansion coefficient is defined and associated with at least one, and in some cases more than one, predefined field variable.

Initial conditions: In all tests the initial value of all relevant field variables is assumed to be zero at all the nodes.

Results and discussion

The results for loading based on field expansion match those obtained from a similar model using thermal expansion. The one-dimensional elements are subjected to field and thermal expansion while fully constrained, and the results have been verified by analytical means.

Input files

fieldexp_s4r.inp	S4R element using a linear elastic material model and loaded with both field and thermal expansion.
fieldexp_sc8r.inp	SC8R element using a linear elastic material model and loaded with field expansion driven by a single field variable. Tests nonlinear static and linear perturbation steps.
fieldexp_m3d4.inp	M3D4R element using a linear elastic material model and loaded with both field and thermal expansion.
buckleplate_s8r5_ fieldexpan_riks.inp	S8R5 element using an elastic material model loaded with field expansion driven by a single field variable. Tests Riks procedure and produces same result as buckleplate_s8r5_riks.inp in *Buckling of a simply supported square plate*.
fieldexp-t2d2-multfld.inp	T2D2 element using a linear elastic material model loaded with both field and thermal expansions. The field expansion behavior is driven by three different field variables. Tests proper interpolation of temperature and predefined field-variable-dependent material data defining field expansion coefficient.
fieldexp-t2d2-reftemp.inp	T2D2 element using a linear elastic material model loaded with both field and thermal expansions. The field expansion behavior is driven by two different field variables. The thermal expansion coefficient and the two field expansion coefficients are assumed to be associated with a nonzero reference temperature and nonzero reference field variable values, respectively.
uexpan1x_field.inp	T2D2 element using a linear elastic material model loaded with field expansion defined using user subroutine *UEXPAN*.

Cohesive element load verification

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

COH2D4 COH2D4P COH2D4T COH3D6 COH3D6P COH3D6T COH3D8 COH3D8P COH3D8T COHAX4 COHAX4P COHAX4T

Features tested

The following features are tested in this verification problem:

- Element-based distributed loading.
- Surface-based distributed loading.
- Thermal loading specified using predefined temperatures.

Problem description

In this verification test all the nodes of each element are fixed, and the reaction forces generated at the nodes as a result of the load application are used to verify the element load calculations. In addition, the effect of thermal loading is verified by allowing each element to deform freely in the thickness direction with the change in temperature. The resulting thermal strains in the thickness direction are compared with the analytical results.

Model:

- COH3D6, COH3D6P, COH3D6T, COH3D8, COH3D8P, and COH3D8T:

Cubic dimensions	$7 \times 7 \times 7$
Thickness	Geometry
Thickness direction	Global 2
Response	Continuum
Centrifugal axis of rotation	(1, 0, 0) through (3.5, −1000, 3.5)
Coriolis axis of rotation	(1, 0, 0) through origin
Gravitational load vector	(0, 1, 0)

- COH2D4, COH2D4P, and COH2D4T:

Planar dimensions	7×7
Thickness	Geometry
Thickness direction	Global 2
Response	Continuum
Centrifugal axis of rotation	(1, 0, 0) through (3.5, −1000, 0.0)

Coriolis axis of rotation	(0, 0, 1) through origin
Gravitational load vector	(0, 1, 0)

- COHAX4, COHAX4P, and COHAX4T:

Planar dimensions	10 × 10
Thickness	Geometry
Thickness direction	Global 1; Global 2
Response	Continuum
Centrifugal axis of rotation	(0, 1, 0) through origin
Gravitational load vector	(0, 1, 0)

Material:

Young's modulus	3×10^6
Poisson's ratio	0.3
Coefficient of thermal expansion	0.0001
Density	10.0

Initial conditions:

Initial temperature	ALL, 0.0

Results and discussion

The calculated reactions are in agreement with the applied loads. In addition, the thermal stresses and strains in the thickness direction match the analytical results for the case of thermal loading.

Input files

Abaqus/Standard input files

coh3d8_loads_std.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, ROTA, P1, P2, P3, P4, P5, P6, P, **TEMPERATURE* for COH3D8.
coh3d6_loads_std.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, ROTA, P1, P2, P3, P4, P5, P6, P, **TEMPERATURE* for COH3D6.
coh2d4_loads_std.inp	BX, BY, GRAV, CENT, CENTRIF, ROTA, P1, P2, P3, P4, P, **TEMPERATURE* for COH2D4.
cohax4_loads_std.inp	BR, BZ, GRAV, CENT, CENTRIF, P1, P2, P3, P4, P, **TEMPERATURE* for COHAX4.
coh3d8p_loads_std.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, ROTA, P1, P2, P3, P4, P5, P6, P, **TEMPERATURE* for COH3D8P.
coh3d6p_loads_std.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, ROTA, P1, P2, P3, P4, P5, P6, P, **TEMPERATURE* for COH3D6P.
coh2d4p_loads_std.inp	BX, BY, GRAV, CENT, CENTRIF, ROTA, P1, P2, P3, P4, P, **TEMPERATURE* for COH2D4P.

cohax4p_loads_std.inp	BR, BZ, GRAV, CENT, CENTRIF, P1, P2, P3, P4, P, *TEMPERATURE for COHAX4P.
coh3d8t_loads_std.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, ROTA, P1, P2, P3, P4, P5, P6, P, temperature boundary condition for COH3D8T.
coh3d6t_loads_std.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, ROTA, P1, P2, P3, P4, P5, P6, P, temperature boundary condition for COH3D6T.
coh2d4t_loads_std.inp	BX, BY, GRAV, CENT, CENTRIF, ROTA, P1, P2, P3, P4, P, temperature boundary condition for COH2D4T.
cohax4t_loads_std.inp	BR, BZ, GRAV, CENT, CENTRIF, P1, P2, P3, P4, P, temperature boundary condition for COHAX4T.
coh_corioload.inp	CORIO for COH3D8, COH3D6, and COH2D4.
cohp_corioload.inp	CORIO for COH3D8P, COH3D6P, and COH2D4P.
coht_corioload.inp	CORIO for COH3D8T, COH3D6T, and COH2D4T.
coh3d8t_thermal_loads.inp	BF, S1, S2, F1, F2, R1, R2 for COH3D8T.
coh3d6t_thermal_loads.inp	BF, S1, S3, F1, F3, R1, R3 for COH3D6T.
coh2d4t_thermal_loads.inp	BF, S1, S2, F1, F2, R1, R2 for COH2D4T.
cohax4t_thermal_loads.inp	BF, S1, S3, F1, F3, R1, R3 for COHAX4T.
coh2d4t_cthermal_loads.inp	*CFILM, *CRADIATE for COH2D4T.

Abaqus/Explicit input files

coh_bf_grav_xpl.inp	BX, BR, BY, BZ, GRAV for COH3D8, COH3D6, COH2D4, and COHAX4.
coh_p_vp_xpl.inp	P1, P2, P3, P, VP1, VP2, VP3, VP4, VP for COH3D8, COH3D6, COH2D4, and COHAX4.
coh_thermal_xpl.inp	*TEMPERATURE for COH3D8, COH3D6, COH2D4, and COHAX4.

ELBOW elements

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

General loading

Problem description

Model:

Length	15.0
Elbow section data	r_o = 10.0, t = 1.0
Centrifugal axis of rotation	(0, 0, 1) through midspan
Gravity load vector	(0, 0, 1)

Material:

Young's modulus	3×10^6
Poisson's ratio	0.3
Density	1.0
Coefficient of thermal expansion	0.0001

Initial conditions:

Initial temperature	ALL, −10.0

General: The boundary condition NODEFORM is used for load types BX, BY, BZ, GRAV, CENT, CENTRIF, and ROTA on ELBOW31 and ELBOW32 elements, preventing cross-sectional deformations.

Results and discussion

The calculated reactions are in agreement with the applied loads.

Input files

exel1xd1.inp	ELBOW31: BX, BY, BZ, GRAV, CENT, PI, CENTRIF, **TEMPERATURE*, perturbation step with **LOAD CASE*.
exel1xdr.inp	ELBOW31: ROTA.
exelbxd1.inp	ELBOW31B: BX, BY, BZ, GRAV, CENT, PI, CENTRIF, **TEMPERATURE*.
exelbxdr.inp	ELBOW31B: ROTA.

exelcxd1.inp	ELBOW31C: BX, BY, BZ, GRAV, CENT, PI, CENTRIF, **TEMPERATURE*.
exelcxdr.inp	ELBOW31C: ROTA.
exel2xd1.inp	ELBOW32: BX, BY, BZ, GRAV, CENT, PI, CENTRIF, **TEMPERATURE*.
exel2xdr.inp	ELBOW32: ROTA.

Closed-end pressure loading

Problem description

Closed-end pressure loading of ELBOW elements is verified. A single element is oriented at 45° to the x- and z-axis in a fluid of density 1×10^3. The magnitude of the acceleration resulting from gravity is 9.8, and the positive hydrostatic pressure gradient is in the negative 3-direction. The test consists of completely constraining all degrees of freedom and recovering the reaction forces under the hydrostatic pressure load on the elbow. According to Archimedes' Principle, the net reaction in the 3-direction should be the buoyant force, which in this case is 2203.04. All other reaction forces and moments should sum to zero. The results also indicate that the directions of the nonzero reaction forces and moments on individual nodes are correct.

The second test is of a curved elbow with closed-end conditions modeled by an ELBOW element. Internal pressure is applied to the elbow. The reaction forces should sum to 0. in all directions.

Model:

Length	1.0
Elbow section data	$r_o = 0.275$, $t = 0.025$
Effective diameter	0.535
Gravity vector	(0, 0, 1)

Material:

Young's modulus	1×10^{10}
Poisson's ratio	0.33

Loading:

Uniform pressure magnitude	10.0

Initial conditions:

Zero pressure height	0.0
Reference pressure height	−2.0
Reference pressure magnitude	1.96×10^4

Results and discussion

The calculated reactions are in agreement with the applied loads.

Input files

exel1xdh.inp	ELBOW31: HPE, PI.
exelbxdh.inp	ELBOW31B: HPE, PI.
exelcxdh.inp	ELBOW31C: HPE, PI.
exel2xdh.inp	ELBOW32: HPE, PI.

Continuum pore pressure elements

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Plane strain elements with pore pressure

Problem description

Model:

Planar dimension	3×5
Gravity load vector	(1, 1, 0)

Material:

Young's modulus	1×10^{8}
Poisson's ratio	0.0
Density	1.4142
Permeability	1×10^{-5}
Specific weight of fluid	1.0

Initial conditions:

Initial void ratio	1.0
Hydrostatic pressure datum	5.0
Hydrostatic pressure elevation	0.0
Sink pore pressure	14.7

Results and discussion

The calculated reactions are in agreement with the applied loads.

Input files

CPE4P element load tests:

ece4pfde.inp	CENTRIF, BX, BY, GRAV, P1, P2, P3, P4, HP1, HP2, HP3, HP4, Q1, Q2, Q3, Q4, S1, S2, S3, S4.
ece4pfdl.inp	P, HP, Q, S.
ece4pfdr.inp	ROTA.

CPE4PH element load tests:

ece4phde.inp — CENTRIF, BX, BY, GRAV, P1, P2, P3, P4, HP1, HP2, HP3, HP4, Q1, Q2, Q3, Q4, S1, S2, S3, S4.

ece4phdl.inp — P, HP, Q, S.

ece4phdr.inp — ROTA.

CPE4RP element load tests:

ece4prde.inp — CENTRIF, BX, BY, GRAV, P1, P2, P3, P4, HP1, HP2, HP3, HP4, Q1, Q2, Q3, Q4, S1, S2, S3, S4.

ece4prdl.inp — P, HP, Q, S.

ece4prdr.inp — ROTA.

CPE4RPH element load tests:

ece4pyde.inp — CENTRIF, BX, BY, GRAV, P1, P2, P3, P4, HP1, HP2, HP3, HP4, Q1, Q2, Q3, Q4, S1, S2, S3, S4.

ece4pydl.inp — P, HP, Q, S.

ece4pydr.inp — ROTA.

CPE6MP element load tests:

ece6pkde.inp — CENTRIF, BX, BY, GRAV, P1, P2, P3, HP1, HP2, HP3, Q1, Q2, Q3, S1, S2, S3.

ece6pkdl.inp — P, HP, Q, S.

ece6pkdr.inp — ROTA.

CPE6MPH element load tests:

ece6plde.inp — CENTRIF, BX, BY, GRAV, P1, P2, P3, HP1, HP2, HP3, Q1, Q2, Q3, S1, S2, S3.

ece6pldl.inp — P, HP, Q, S.

ece6pldr.inp — ROTA.

CPE8P element load tests:

ece8pfde.inp — CENTRIF, BX, BY, GRAV, P1, P2, P3, P4, HP1, HP2, HP3, HP4, Q1, Q2, Q3, Q4, Q1D, Q2D, Q3D, Q4D, S1, S2, S3, S4.

ece8pfdl.inp — P, HP, Q, QD, S.

ece8pfdr.inp — ROTA.

CPE8PH element load tests:

ece8phde.inp — CENTRIF, BX, BY, GRAV, P1, P2, P3, P4, HP1, HP2, HP3, HP4, Q1, Q2, Q3, Q4, Q1D, Q2D, Q3D, Q4D, S1, S2, S3, S4.

ece8phdl.inp — P, HP, Q, QD, S.

ece8phdr.inp — ROTA.

CPE8RP element load tests:

ece8prde.inp — CENTRIF, BX, BY, GRAV, P1, P2, P3, P4, HP1, HP2, HP3, HP4, Q1, Q2, Q3, Q4, Q1D, Q2D, Q3D, Q4D, S1, S2, S3, S4.

ece8prdl.inp — P, HP, Q, QD, S.

ece8prdr.inp — ROTA.

CPE8RPH element load tests:

ece8pyde.inp	CENTRIF, BX, BY, GRAV, P1, P2, P3, P4, HP1, HP2, HP3, HP4, Q1, Q2, Q3, Q4, Q1D, Q2D, Q3D, Q4D, S1, S2, S3, S4.
ece8pydl.inp	P, HP, Q, QD, S.
ece8pydr.inp	ROTA.

Axisymmetric elements with pore pressure

Problem description

Model:

Planar dimension	3×5
Inside radius	1.0
Gravity load vector	(1, 1, 0)

Material:

Young's modulus	1×10^{8}
Poisson's ratio	0.0
Density	1.4142
Permeability	1×10^{-5}
Specific weight of fluid	1.0

Initial conditions:

Initial void ratio	1.0
Hydrostatic pressure datum	5.0
Hydrostatic pressure elevation	0.0
Sink pore pressure	14.7

Results and discussion

The calculated reactions are in agreement with the applied loads.

Input files

CAX4P element load tests:

eca4pfde.inp	CENTRIF, BR, BZ, GRAV, P1, P2, P3, P4, HP1, HP2, HP3, HP4, Q1, Q2, Q3, Q4, S1, S2, S3, S4.
eca4pfdl.inp	P, HP, Q, S.

CAX4PH element load tests:

eca4phde.inp	CENTRIF, BR, BZ, GRAV, P1, P2, P3, P4, HP1, HP2, HP3, HP4, Q1, Q2, Q3, Q4, S1, S2, S3, S4.
eca4phdl.inp	P, HP, Q, S.

CAX4RP element load tests:

eca4prde.inp	CENTRIF, BR, BZ, GRAV, P1, P2, P3, P4, HP1, HP2, HP3, HP4, Q1, Q2, Q3, Q4, S1, S2, S3, S4.
eca4prdl.inp	P, HP, Q, S.

CAX4RPH element load tests:

eca4pyde.inp	CENTRIF, BR, BZ, GRAV, P1, P2, P3, P4, HP1, HP2, HP3, HP4, Q1, Q2, Q3, Q4, S1, S2, S3, S4.
eca4pydl.inp	P, HP, Q, S.

CAX6MP element load tests:

eca6pkde.inp	CENTRIF, BR, BZ, GRAV, P1, P2, P3, HP1, HP2, HP3, Q1, Q2, Q3, S1, S2, S3.
eca6pkdl.inp	P, HP, Q, S.

CAX6MPH element load tests:

eca6plde.inp	CENTRIF, BR, BZ, GRAV, P1, P2, P3, HP1, HP2, HP3, Q1, Q2, Q3, S1, S2, S3.
eca6pldl.inp	P, HP, Q, S.

CAX8P element load tests:

eca8pfde.inp	CENTRIF, BR, BZ, GRAV, P1, P2, P3, P4, HP1, HP2, HP3, HP4, Q1, Q2, Q3, Q4, Q1D, Q2D, Q3D, Q4D, S1, S2, S3, S4.
eca8pfdl.inp	P, HP, Q, QD, S.

CAX8PH element load tests:

eca8phde.inp	CENTRIF, BR, BZ, GRAV, P1, P2, P3, P4, HP1, HP2, HP3, HP4, Q1, Q2, Q3, Q4, Q1D, Q2D, Q3D, Q4D, S1, S2, S3, S4.
eca8phdl.inp	P, HP, Q, QD, S.

CAX8RP element load tests:

eca8prde.inp	CENTRIF, BR, BZ, GRAV, P1, P2, P3, P4, HP1, HP2, HP3, HP4, Q1, Q2, Q3, Q4, Q1D, Q2D, Q3D, Q4D, S1, S2, S3, S4.
eca8prdl.inp	P, HP, Q, QD, S.

CAX8RPH element load tests:

eca8pyde.inp	CENTRIF, BR, BZ, GRAV, P1, P2, P3, P4, HP1, HP2, HP3, HP4, Q1, Q2, Q3, Q4, Q1D, Q2D, Q3D, Q4D, S1, S2, S3, S4.
eca8pydl.inp	P, HP, Q, QD, S.

Three-dimensional elements with pore pressure

Problem description

Model:

Cubic dimension	$3 \times 5 \times 1$
Gravity load vector	(1, 1, 1)

Material:

Young's modulus	1×10^8
Poisson's ratio	0.0
Density	1.7321
Permeability	1×10^{-5}
Specific weight of fluid	1.0

Initial conditions:

Initial void ratio	1.0
Hydrostatic pressure datum	5.0
Hydrostatic pressure elevation	0.0
Sink pore pressure	14.7

Results and discussion

The calculated reactions are in agreement with the applied loads.

Input files

C3D4P element load tests:

elem_load_c3d4p_1.inp	CENTRIF, BX, BY, BZ, GRAV, P1, P2, P3, P4, HP1, HP2, HP3, HP4, Q1, Q2, Q3, Q4, S1, S2, S3, S4.
elem_load_c3d4p_2.inp	P, HP, Q, S.
elem_load_c3d4p_3.inp	ROTA.

C3D6P element load tests:

elem_load_c3d6p_1.inp	CENTRIF, BX, BY, BZ, GRAV, P1, P2, P3, P4, P5, HP1, HP2, HP3, HP4, HP5, Q1, Q2, Q3, Q4, Q5, S1, S2, S3, S4, S5.
elem_load_c3d6p_2.inp	P, HP, Q, S.
elem_load_c3d6p_3.inp	ROTA.

C3D8P element load tests:

ec38pfde.inp	CENTRIF, BX, BY, BZ, GRAV, P1, P2, P3, P4, P5, P6, HP1, HP2, HP3, HP4, HP5, HP6, Q1, Q2, Q3, Q4, Q5, Q6, S1, S2, S3, S4, S5, S6.
ec38pfdl.inp	P, HP, Q, S.
ec38pfdr.inp	ROTA.

C3D8PH element load tests:

ec38phde.inp	CENTRIF, BX, BY, BZ, GRAV, P1, P2, P3, P4, P5, P6, HP1, HP2, HP3, HP4, HP5, HP6, Q1, Q2, Q3, Q4, Q5, Q6, S1, S2, S3, S4, S5, S6.
ec38phdl.inp	P, HP, Q, S.
ec38phdr.inp	ROTA.

C3D8RP element load tests:

ec38prde.inp	CENTRIF, BX, BY, BZ, GRAV, P1, P2, P3, P4, P5, P6, HP1, HP2, HP3, HP4, HP5, HP6, Q1, Q2, Q3, Q4, Q5, Q6, S1, S2, S3, S4, S5, S6.
ec38prdl.inp	P, HP, Q, S.

ec38prdr.inp	ROTA.

C3D8RPH element load tests:

ec38pyde.inp	CENTRIF, BX, BY, BZ, GRAV, P1, P2, P3, P4, P5, P6, HP1, HP2, HP3, HP4, HP5, HP6, Q1, Q2, Q3, Q4, Q5, Q6, S1, S2, S3, S4, S5, S6.
ec38pydl.inp	P, HP, Q, S.
ec38pydr.inp	ROTA.

C3D10MP element load tests:

ec3apkde.inp	CENTRIF, BX, BY, BZ, GRAV, P1, P2, P3, P4, HP1, HP2, HP3, HP4, Q1, Q2, Q3, Q4, S1, S2, S3, S4.
ec3apkdl.inp	P, HP, Q, S.
ec3apkdr.inp	ROTA.

C3D10MPH element load tests:

ec3aplde.inp	CENTRIF, BX, BY, BZ, GRAV, P1, P2, P3, P4, HP1, HP2, HP3, HP4, Q1, Q2, Q3, Q4, S1, S2, S3, S4.
ec3apldl.inp	P, HP, Q, S.
ec3apldr.inp	ROTA.

C3D20P element load tests:

ec3kpfde.inp	CENTRIF, BX, BY, BZ, GRAV, P1, P2, P3, P4, P5, P6, HP1, HP2, HP3, HP4, HP5, HP6, Q1, Q2, Q3, Q4, Q5, Q6, Q1D, Q2D, Q3D, Q4D, Q5D, Q6D, S1, S2, S3, S4, S5, S6.
ec3kpfde.f	User subroutines *FLOW* and *DFLOW* used in ec3kpfde.inp.
ec3kpfdl.inp	P, HP, Q, QD, S.
ec3kpfdr.inp	ROTA.

C3D20PH element load tests:

ec3kphde.inp	CENTRIF, BX, BY, BZ, GRAV, P1, P2, P3, P4, P5, P6, HP1, HP2, HP3, HP4, HP5, HP6, Q1, Q2, Q3, Q4, Q5, Q6, Q1D, Q2D, Q3D, Q4D, Q5D, Q6D, S1, S2, S3, S4, S5, S6.
ec3kphdl.inp	P, HP, Q, QD, S.
ec3kphdr.inp	ROTA.

C3D20RP element load tests:

ec3kprde.inp	CENTRIF, BX, BY, BZ, GRAV, P1, P2, P3, P4, P5, P6, HP1, HP2, HP3, HP4, HP5, HP6, Q1, Q2, Q3, Q4, Q5, Q6, Q1D, Q2D, Q3D, Q4D, Q5D, Q6D, S1, S2, S3, S4, S5, S6.
ec3kprdl.inp	P, HP, Q, QD, S.
ec3kprdr.inp	ROTA.

C3D20RPH element load tests:

ec3kpyde.inp	CENTRIF, BX, BY, BZ, GRAV, P1, P2, P3, P4, P5, P6, HP1, HP2, HP3, HP4, HP5, HP6, Q1, Q2, Q3, Q4, Q5, Q6, Q1D, Q2D, Q3D, Q4D, Q5D, Q6D, S1, S2, S3, S4, S5, S6.
ec3kpydl.inp	P, HP, Q, QD, S.
ec3kpydr.inp	ROTA.

CAXA elements with pore pressure

Problem description

Model:

Planar dimension	3×5
Inside radius	1.0
Gravity load vector	(1, 1, 0)

Material:

Young's modulus	1×10^{8}
Poisson's ratio	0.0
Density	1.4142
Permeability	1×10^{-5}
Specific weight of fluid	1.0

Initial conditions:

Initial void ratio	1.0
Hydrostatic pressure datum	5.0
Hydrostatic pressure elevation	0.0
Sink pore pressure	14.7

Results and discussion

The calculated reactions are in agreement with the applied loads.

Input files

ecnwpfde.inp	CAXA8P1: BX, BZ, GRAV, P1, P2, P3, P4, HP1, HP2, HP3, HP4, Q1, Q2, Q3, Q4, Q1D, Q2D, Q3D, Q4D, S1, S2, S3, S4.
ecnwprde.inp	CAXA8RP1: BX, BZ, GRAV, P1, P2, P3, P4, HP1, HP2, HP3, HP4, Q1, Q2, Q3, Q4, Q1D, Q2D, Q3D, Q4D, S1, S2, S3, S4.
ecnxpfde.inp	CAXA8P2: BX, BZ, GRAV, P1, P2, P3, P4, HP1, HP2, HP3, HP4, Q1, Q2, Q3, Q4, Q1D, Q2D, Q3D, Q4D, S1, S2, S3, S4.
ecnxprde.inp	CAXA8RP2: BX, BZ, GRAV, P1, P2, P3, P4, HP1, HP2, HP3, HP4, Q1, Q2, Q3, Q4, Q1D, Q2D, Q3D, Q4D, S1, S2, S3, S4.
ecnypfde.inp	CAXA8P3: BX, BZ, GRAV, P1, P2, P3, P4, HP1, HP2, HP3, HP4, Q1, Q2, Q3, Q4, Q1D, Q2D, Q3D, Q4D, S1, S2, S3, S4.
ecnyprde.inp	CAXA8RP3: BX, BZ, GRAV, P1, P2, P3, P4, HP1, HP2, HP3, HP4, Q1, Q2, Q3, Q4, Q1D, Q2D, Q3D, Q4D, S1, S2, S3, S4.
ecnzpfde.inp	CAXA8P4: BX, BZ, GRAV, P1, P2, P3, P4, HP1, HP2, HP3, HP4, Q1, Q2, Q3, Q4, Q1D, Q2D, Q3D, Q4D, S1, S2, S3, S4.
ecnzprde.inp	CAXA8RP4: BX, BZ, GRAV, P1, P2, P3, P4, HP1, HP2, HP3, HP4, Q1, Q2, Q3, Q4, Q1D, Q2D, Q3D, Q4D, S1, S2, S3, S4.

Plane strain pore-thermal elements

Problem description

Model:

Planar dimension	3×5
Gravity direction	(1, 1, 0)

Material:

Young's modulus	1×10^{8}
Poisson's ratio	0.0
Density	1.4142
Permeability	1×10^{-5}
Thermal conductivity	0.1

Initial conditions:

Initial void ratio	1.0
Initial temperature	0.0
Initial pore pressure	0.0

Results and discussion

The calculated reactions are in agreement with the applied loads.

Input files

cpe4pt_mech_loads.inp	CPE4PT: BX, BY, GRAV, CENTRIF, P1, P2, P3, P4, HP1, HP2, HP3, HP4.
cpe4pt_surf_mech_loads.inp	CPE4PT: P, HP.
cpe4rpt_mech_loads.inp	CPE4RPT: BX, BY, GRAV, CENTRIF, P1, P2, P3, P4, HP1, HP2, HP3, HP4.
cpe4rpt_surf_mech_loads.inp	CPE4RPT: P, HP.
cpe4pht_mech_loads.inp	CPE4PHT: BX, BY, GRAV, CENTRIF, P1, P2, P3, P4, HP1, HP2, HP3, HP4.
cpe4pht_surf_mech_loads.inp	CPE4PHT: P, HP.
cpe4rpht_mech_loads.inp	CPE4RPHT: BX, BY, GRAV, CENTRIF, P1, P2, P3, P4, HP1, HP2, HP3, HP4.
cpe4rpht_surf_mech_loads.inp	CPE4RPHT: P, HP.

Plane strain pore-thermal elements with flow loads

Problem description

Model:

Planar dimension	3×5

Material:

Young's modulus	1×10^{8}
Poisson's ratio	0.0
Density	1.4142
Permeability	1×10^{-5}
Thermal conductivity	0.1
Specific heat	0.39

Initial conditions:

Initial void ratio	1.0
Initial temperature	0.0
Initial pore pressure	0.0

Results and discussion

The calculated reactions are in agreement with the applied loads.

Input files

cpe4pt_flow_loads.inp	CPE4PT: Q1, Q2, Q3, Q4, S1, S2, S3, S4.
cpe4pt_surf_flow_loads.inp	CPE4PT: Q, S.
cpe4rpt_flow_loads.inp	CPE4RPT: Q1, Q2, Q3, Q4, S1, S2, S3, S4.
cpe4rpt_surf_flow_loads.inp	CPE4RPT: Q, S.
cpe4pht_flow_loads.inp	CPE4PHT: Q1, Q2, Q3, Q4, S1, S2, S3, S4.
cpe4pht_surf_flow_loads.inp	CPE4PHT: Q, S.
cpe4rpht_flow_loads.inp	CPE4RPHT: Q1, Q2, Q3, Q4, S1, S2, S3, S4.
cpe4rpht_surf_flow_loads.inp	CPE4RPHT: Q, S.

Plane strain pore-thermal elements with heat loads

Problem description

Model:

Planar dimension	7×7

Material:

Young's modulus	30×10^6
Poisson's ratio	0.3
Permeability	1×10^{-5}
Coefficient of thermal expansion	0.0
Thermal conductivity	3.77×10^{-5}
Density	82.9
Specific heat	0.39
Coefficient of thermal expansion, pore fluid	0.0
Thermal conductivity, pore fluid	3.77×10^{-5}
Density, pore fluid	82.9
Specific heat, pore fluid	0.39

Initial conditions:

Initial void ratio	1.0
Initial temperature	0.0
Initial pore pressure	0.0

Results and discussion

The calculated reactions are in agreement with the applied loads.

Input files

cpe4pt_heat_loads.inp	CPE4PT: BF, S1, S2, S3, S4, F1, F2, F3, F4, R1, R2, R3, R4.
cpe4pt_surf_heat_loads.inp	CPE4PT: S, F, R.
cpe4rpt_heat_loads.inp	CPE4RPT: BF, S1, S2, S3, S4, F1, F2, F3, F4, R1, R2, R3, R4.
cpe4rpt_surf_heat_loads.inp	CPE4RPT: S, F, R.
cpe4pht_heat_loads.inp	CPE4PHT: BF, S1, S2, S3, S4, F1, F2, F3, F4, R1, R2, R3, R4.
cpe4pht_surf_heat_loads.inp	CPE4PHT: S, F, R.
cpe4rpht_heat_loads.inp	CPE4RPHT: BF, S1, S2, S3, S4, F1, F2, F3, F4, R1, R2, R3, R4.
cpe4rpht_surf_heat_loads.inp	CPE4RPHT: S, F, R.

Three-dimensional pore-thermal elements

Problem description

Model:

Cubic dimension	$7 \times 7 \times 7$
Gravity direction	(1, 0, 0)

Material:

Modulus	3×10^6
Density	10.0
Expansion	0.0001
Specific heat	1.0
Conductivity	0.1
Density, pore fluid	10.0
Expansion, pore fluid	0.0001
Specific heat, pore fluid	1.0
Conductivity, pore fluid	0.1
Permeability	0.01
Specific weight of fluid	1.0

Initial conditions:

Initial void ratio	1.0
Initial temperature	0.0
Initial pore pressure	0.0

Results and discussion

The calculated reactions are in agreement with the applied loads.

Input files

c3d4pt_loads.inp	C3D4PT: BX, BY, BZ, GRAV, CENT, CENTRIF, P1, P2, P3, P4, P5, P6, HP1, HP2, HP3, HP4, HP5, HP6.
c3d8pt_loads.inp	C3D8PT: BX, BY, BZ, GRAV, CENT, CENTRIF, P1, P2, P3, P4, P5, P6, HP1, HP2, HP3, HP4, HP5, HP6.
c3d8rpt_loads.inp	C3D8RPT: BX, BY, BZ, GRAV, CENT, CENTRIF, P1, P2, P3, P4, P5, P6, HP1, HP2, HP3, HP4, HP5, HP6.
c3d8rpht_loads.inp	C3D8RPHT: BX, BY, BZ, GRAV, CENT, CENTRIF, P1, P2, P3, P4, P5, P6, HP1, HP2, HP3, HP4, HP5, HP6.
c3d10pt_loads.inp	C3D10PT: BX, BY, BZ, GRAV, CENT, CENTRIF, P1, P2, P3, P4, HP1, HP2, HP3, HP4.
c3d10mpt_loads.inp	C3D10MPT: BX, BY, BZ, GRAV, CENT, CENTRIF, P1, P2, P3, P4, HP1, HP2, HP3, HP4.

Three-dimensional pore-thermal elements with flow loads

Problem description

Model:

Cubic dimension	$5 \times 3 \times 1$

Gravity direction	(1, 1, 1)

Material:

Modulus	1×10^{8}
Density	1.7321
Expansion	0.0
Specific heat	10.0
Conductivity	1.0
Density, pore fluid	1.7321
Expansion, pore fluid	0.0
Specific heat, pore fluid	10.0
Conductivity, pore fluid	1.0
Permeability	1×10^{-5}
Specific weight of fluid	1.0

Initial conditions:

Initial void's ratio	1.0
Initial temperature	0.0
Initial pore pressure	0.0

Results and discussion

The calculated reactions are in agreement with the applied loads.

Input files

c3d4ph_dsflow_loads.inp	C3D4PH: P, HP, Q, S.
c3d4pt_dsflow_loads.inp	C3D4PT: P, HP, Q, S.
c3d8pt_dflow_loads.inp	C3D8PT: BX, BY, BZ, GRAV, CENT, CENTRIF, P1, P2, P3, P4, P5, P6, HP1, HP2, HP3, HP4, HP5, HP6, Q1, Q2, Q3, Q4, Q5, Q6, S1, S2, S3, S4, S5, S6.
c3d8pt_sflow_loads.inp	C3D8PT: P, HP, Q, S.
c3d8rpt_dflow_loads.inp	C3D8RPT: BX, BY, BZ, GRAV, CENT, CENTRIF, P1, P2, P3, P4, P5, P6, HP1, HP2, HP3, HP4, HP5, HP6, Q1, Q2, Q3, Q4, Q5, Q6, S1, S2, S3, S4, S5, S6.
c3d8rpht_dflow_loads.inp	C3D8RPHT: BX, BY, BZ, GRAV, CENT, CENTRIF, P1, P2, P3, P4, P5, P6, HP1, HP2, HP3, HP4, HP5, HP6, Q1, Q2, Q3, Q4, Q5, Q6, S1, S2, S3, S4, S5, S6.
c3d10mpt_flow_loads.inp	C3D10MPT: BX, BY, BZ, GRAV, CENT, CENTRIF, P1, P2, P3, P4, HP1, HP2, HP3, HP4, Q1, Q2, Q3, Q4, S1, S2, S3, S4.
c3d10p_dsflow_loads.inp	C3D10P: P, HPQ, S.
c3d10pt_dsflow_loads.inp	C3D10PT: P, HPQ, S.
c3d10mpt_dsflow_loads.inp	C3D10MPT: P, HPQ, S.

Three-dimensional pore-thermal elements with heat loads

Problem description

Model: For this set of verification problems both the solid and the pore fluid used identical heat transfer properties so that results could be compared with conventional heat transfer elements.

Cubic dimension	$7 \times 7 \times 7$

Material:

Modulus	3×10^6
Density	82.9
Expansion	0.0
Specific heat	0.39
Conductivity	3.77×10^{-5}
Density, pore fluid	82.9
Expansion, pore fluid	0.0
Specific heat, pore fluid	0.39
Conductivity, pore fluid	3.77×10^{-5}
Permeability	0.001
Specific weight of fluid	10.0

Initial conditions:

Initial void's ratio	1.0
Initial temperature	0.0
Initial pore pressure	0.0

Results and discussion

The calculated reactions are in agreement with the applied loads.

Input files

c3d4pt_heat_loads.inp	C3D4PT: S, F, R.
c3d8pt_heat_loads.inp	C3D8PT: S, F, R.
c3d8rpt_heat_loads.inp	C3D8RPT: S, F, R.
c3d8rpht_heat_loads.inp	C3D8RPHT: S, F, R.
c3d10pt_heat_loads.inp	C3D10PT: S, F, R.
c3d10mpt_heat_loads.inp	C3D10MPT: S, F, R.

Axisymmetric pore-thermal elements

Problem description

Model:

Cubic dimension	3×3

Material:

Modulus	3×10^{6}
Density	5×10^{-5}
Expansion	0.0001
Specific heat	1.0
Conductivity	0.1
Density, pore fluid	5×10^{5}
Expansion, pore fluid	0.0001
Specific heat, pore fluid	1.0
Conductivity, pore fluid	0.1
Permeability	0.01
Specific weight of fluid	1.0

Initial conditions:

Initial void's ratio	1.0
Initial temperature	0.0
Initial pore pressure	0.0

Results and discussion

The calculated reactions are in agreement with the applied loads.

Input files

cax4pt_loads.inp	CAX4PT: CENT, CENTRIF, BR, GRAV,HP1, HP2, HP3, HP4, P1, P2, P3, P4.
cax4rpt_loads.inp	CAX4RPT: CENT, CENTRIF, BR, GRAV,HP1, HP2, HP3, HP4, P1, P2, P3, P4.
cax4rpht_loads.inp	CAX4RPHT: CENT, CENTRIF, BR, GRAV,HP1, HP2, HP3, HP4, P1, P2, P3, P4.

Axisymmetric pore-thermal elements with flow loads

Problem description

Model:

Cubic dimension	3×5

Material:

Modulus	1×10^{8}
Density	1.4142
Expansion	0.0
Specific heat	10.0
Conductivity	1.0
Density, pore fluid	1.4142
Expansion, pore fluid	0.0
Specific heat, pore fluid	10.0
Conductivity, pore fluid	1.0
Permeability	1×10^{-5}
Specific weight of fluid	1.0

Initial conditions:

Initial void's ratio	1.0
Initial temperature	0.0
Initial pore pressure	0.0

Results and discussion

The calculated reactions are in agreement with the applied loads.

Input files

cax4pt_dflow_loads.inp	CAX4PT: CENTRIF, BR,HP1, HP2, HP3, HP4, Q1, Q2, Q3, Q4, S1, S2, S3, S4.
cax4rpt_dflow_loads.inp	CAX4RPT: CENTRIF, BR,HP1, HP2, HP3, HP4, Q1, Q2, Q3, Q4, S1, S2, S3, S4.
cax4rpht_dflow_loads.inp	CAX4RPHT: CENTRIF, BR,HP1, HP2, HP3, HP4, Q1, Q2, Q3, Q4, S1, S2, S3, S4.

Axisymmetric pore-thermal elements with heat loads

Problem description

Model: For this set of verification problems both the solid and the pore fluid used identical heat transfer properties so that results could be compared with conventional heat transfer elements.

Cubic dimension	7×7

Material:

Modulus	30×10^6
Density	82.9
Expansion	0.0
Specific heat	0.39
Conductivity	3.77×10^{-5}
Density, pore fluid	82.9
Expansion, pore fluid	0.0
Specific heat, pore fluid	0.39
Conductivity, pore fluid	3.77×10^{-5}
Permeability	0.001
Specific weight of fluid	10.0

Initial conditions:

Initial void's ratio	1.0
Initial temperature	0.0
Initial pore pressure	0.0

Results and discussion

The calculated reactions are in agreement with the applied loads.

Input files

cax4pt_heat_loads.inp	CAX4PT: S, F, R.
cax4rpt_heat_loads.inp	CAX4RPT: S, F, R.
cax4rpht_heat_loads.inp	CAX4RPHT: S, F, R.

Continuum and shell heat transfer elements

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

One-dimensional heat transfer elements

Problem description

Model:

Length	7.0
Area	3.0

The area of element types DCCAX2 and DCCAX2D is 0.4774648.

Material:

Thermal conductivity	3.77×10^{-5}
Sink (bulk fluid) temperature	75.0
Absolute zero temperature	–460.0
Specific heat	0.39
Density	82.9

Results and discussion

The calculated reactions are in agreement with the applied loads.

Input files

ec12dfdc.inp	DC1D2: BF, F1, F2, R1, R2, S1, S2.
ec13dfdc.inp	DC1D3: BF, F1, F2, R1, R2, S1, S2.
ec12dcdc.inp	DCC1D2: BF, F1, F2, R1, R2, S1, S2.
ec12dddc.inp	DCC1D2D: BF, F1, F2, R1, R2, S1, S2.
eca2dcdc.inp	DCCAX2: BF, F1, F2, R1, R2, S1, S2.
eca2dddc.inp	DCCAX2D: BF, F1, F2, R1, R2, S1, S2.

Planar solid heat transfer elements

Problem description

Model:

Square dimension	7×7

Thickness	1.0

Material:

Thermal conductivity	3.77×10^{-5}
Sink (bulk fluid) temperature	75.0
Absolute zero temperature	−460.0
Specific heat	0.39
Density	82.9

Results and discussion

The calculated reactions are in agreement with the applied loads.

Input files

DC2D3 element load tests:

ec23dfdc.inp	BF, F1, F2, F3, R1, R2, R3, S1, S2, S3.
ec23dfdj.inp	F, R, S.

DC2D4 element load tests:

ec24dfdc.inp	BF, F1, F2, F3, F4, R1, R2, R3, R4, S1, S2, S3, S4.
ec24dfdj.inp	F, R, S.

DC2D6 element load tests:

ec26dfdc.inp	BF, F1, F2, F3, R1, R2, R3, S1, S2, S3.
ec26dfdj.inp	F, R, S.

DC2D8 element load tests:

ec28dfdc.inp	BF, F1, F2, F3, F4, R1, R2, R3, R4, S1, S2, S3, S4.
ec28dfdj.inp	F, R, S.

DCC2D4 element load tests:

ec24dcdc.inp	BF, F1, F2, F3, F4, R1, R2, R3, R4, S1, S2, S3, S4.
ec24dcdj.inp	F, R, S.

DCC2D4D element load tests:

ec24dddc.inp	BF, F1, F2, F3, F4, R1, R2, R3, R4, S1, S2, S3, S4.
ec24dddj.inp	F, R, S.

Axisymmetric solid heat transfer elements

Problem description

Model:

Planar dimensions	7 × 7

Inside radius	1.0

Material:

Thermal conductivity	3.77×10^{-5}
Sink (bulk fluid) temperature	75.0
Absolute zero temperature	–460.0
Specific heat	0.39
Density	82.9

Results and discussion

The calculated reactions are in agreement with the applied loads.

Input files

DCAX3 element load tests:

eca3dfdc.inp	BF, F1, F2, F3, R1, R2, R3, S1, S2, S3.
eca3dfdj.inp	F, R, S.

DCAX4 element load tests:

eca4dfdc.inp	BF, F1, F2, F3, F4, R1, R2, R3, R4, S1, S2, S3, S4.
eca4dfdj.inp	F, R, S.

DCAX6 element load tests:

eca6dfdc.inp	BF, F1, F2, F3, R1, R2, R3, S1, S2, S3.
eca6dfdj.inp	F, R, S.

DCAX8 element load tests:

eca8dfdc.inp	BF, F1, F2, F3, F4, R1, R2, R3, R4, S1, S2, S3, S4.
eca8dfdj.inp	F, R, S.

DCCAX4 element load tests:

eca4dcdc.inp	BF, F1, F2, F3, F4, R1, R2, R3, R4, S1, S2, S3, S4.
eca4dcdj.inp	F, R, S.

DCCAX4D element load tests:

eca4dddc.inp	BF, F1, F2, F3, F4, R1, R2, R3, R4, S1, S2, S3, S4.
eca4dddj.inp	F, R, S.

Three-dimensional solid heat transfer elements

Problem description

Model:

Cubic dimensions	$7 \times 7 \times 7$

Material:

Thermal conductivity	3.77×10^{-5}
Sink (bulk fluid) temperature	75.0
Absolute zero temperature	–460.0
Specific heat	0.39
Density	82.9

Results and discussion

The calculated reactions are in agreement with the applied loads.

Input files

DC3D4 element load tests:

ec34dfdc.inp	BF, F1, F2, F3, F4, R1, R2, R3, R4, S1, S2, S3, S4.
ec34dfdj.inp	AVG, F, R, S.

DC3D6 element load tests:

ec36dfdc.inp	BF, F1, F2, F3, F4, F5, R1, R2, R3, R4, R5, S1, S2, S3, S4, S5.
ec36dfdj.inp	AVG, F, R, S.

DC3D8 element load tests:

ec38dfdc.inp	BF, F1, F2, F3, F4, F5, F6, R1, R2, R3, R4, R5, R6, S1, S2, S3, S4, S5, S6.
ec38dfdj.inp	AVG, F, R, S.

DC3D10 element load tests:

ec3adfdc.inp	BF, F1, F2, F3, F4, R1, R2, R3, R4, S1, S2, S3, S4.
ec3adfdj.inp	AVG, F, R, S.

DC3D15 element load tests:

ec3fdfdc.inp	BF, F1, F2, F3, F4, F5, R1, R2, R3, R4, R5, S1, S2, S3, S4, S5.
ec3fdfdj.inp	AVG, F, R, S.

DC3D20 element load tests:

ec3kdfdc.inp	BF, F1, F2, F3, F4, F5, F6, R1, R2, R3, R4, R5, R6, S1, S2, S3, S4, S5, S6.
ec3kdfdj.inp	AVG, F, R, S.

DCC3D8 element load tests:

ec38dcdc.inp	BF, F1, F2, F3, F4, F5, F6, R1, R2, R3, R4, R5, R6, S1, S2, S3, S4, S5, S6.
ec38dcdj.inp	F, R, S.

DCC3D8D element load tests:

ec38dddc.inp	BF, F1, F2, F3, F4, F5, F6, R1, R2, R3, R4, R5, R6, S1, S2, S3, S4, S5, S6.
ec38dddj.inp	F, R, S.

Axisymmetric heat transfer shell elements

Problem description

Model:

Length	10.0
Radius	5.0
Thickness	0.5

Material:

Thermal conductivity	3.77×10^{-5}
Sink (bulk fluid) temperature	75.0
Absolute zero temperature	–460.0
Specific heat	0.39
Density	82.9

Results and discussion

The calculated reactions are in agreement with the applied loads.

Input files

DSAX1 element load tests:

esa2dxdc.inp	BF, FNEG, FPOS, RNEG, RPOS, SNEG, SPOS.
esa2dxdj.inp	F, R, S.

DSAX2 element load tests:

esa3dxdc.inp	BF, FNEG, FPOS, RNEG, RPOS, SNEG, SPOS.
esa3dxdj.inp	F, R, S.

General heat transfer shell elements

Problem description

Model:

Planar dimensions	7×7
Thickness	0.5

Material:

Thermal conductivity	3.77×10^{-5}
Sink (bulk fluid) temperature	75.0
Absolute zero temperature	–460.0

Specific heat	0.39
Density	82.9

Results and discussion

The calculated reactions are in agreement with the applied loads.

Input files

DS3 element load tests:

es33dxdc.inp	BF, FNEG, FPOS, RNEG, RPOS, SNEG, SPOS.
es33dxdj.inp	F, R, S.

DS4 element load tests:

es34dxdc.inp	BF, FNEG, FPOS, RNEG, RPOS, SNEG, SPOS.
es34dxdj.inp	F, R, S.

DS6 element load tests:

es36dxdc.inp	BF, FNEG, FPOS, RNEG, RPOS, SNEG, SPOS.
es36dxdj.inp	F, R, S.

DS8 element load tests:

es38dxdc.inp	BF, FNEG, FPOS, RNEG, RPOS, SNEG, SPOS.
es38dxdj.inp	F, R, S.

Coupled temperature-displacement elements

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Coupled temperature-displacement truss elements

Problem description

Model:

Length	7.0
Area	3.0
Centrifugal axis of rotation	(0, 1, 0) through origin
Coriolis axis of rotation	(0, 0, 1) through origin (2D)
	(0, 1, 0) through origin (3D)
Gravity load vector	(0, −1, 0)

Material:

Young's modulus	30×10^{6}
Coefficient of thermal expansion	0.0
Thermal conductivity	3.77×10^{-5}
Density	82.9
Sink (bulk fluid) temperature	75.0
Absolute zero temperature	–460.0

Initial conditions:

Initial velocity	ALL, 1, 5.0
	ALL, 2, 2.0

Results and discussion

The calculated reactions are in agreement with the applied loads.

Input files

T2D2T element load tests:

et22tfdd.inp	BX, BY, BF, GRAV, CENT, CENTRIF, CORIO, F1, F2, R1, R2, S1, S2.
et22tfdr.inp	ROTA.

T2D3T element load tests:

et23tfdd.inp	BX, BY, BF, GRAV, CENT, CENTRIF, CORIO, F1, F2, R1, R2, S1, S2.
et23tfdr.inp	ROTA.

T3D2T element load tests:

et32tfdd.inp	BX, BY,BZ, BF, GRAV, CENT, CENTRIF, CORIO, F1, F2, R1, R2, S1, S2.
et32tfdd_po.inp	**POST OUTPUT* analysis.
et32tfdr.inp	ROTA.

T3D3T element load tests:

et33tfdd.inp	BX, BY, BZ, BF, GRAV, CENT, CENTRIF, CORIO, F1, F2, R1, R2, S1, S2.
et33tfdr.inp	ROTA.

Coupled temperature-displacement plane stress, plane strain and generalized plane strain elements

Problem description

Model:

Square dimensions	7×7
Thickness	1.0
Centrifugal axis of rotation	(0, 1, 0) through origin
Coriolis axis of rotation	(0, 0, 1) through origin
Gravity load vector	$(0, -1, 0)$

Material:

Young's modulus	30×10^6
Poisson's ratio	0.3
Coefficient of thermal expansion	0.0
Thermal conductivity	3.77×10^{-5}
Density	82.9
Sink (bulk fluid) temperature	75.0
Absolute zero temperature	−460.0

Initial conditions:

Initial velocity	ALL, 1, 10
	ALL, 2, 5
Hydrostatic pressure datum	3.0
Hydrostatic pressure elevation	0.0

Results and discussion

The calculated reactions are in agreement with the applied loads.

Input files

Abaqus/Standard input files

CPE3T element load tests:

coupledtempload_std_cpe3t.inp	BX, BY, BF, GRAV, F1, F2, F3, P1, P2, P3, R1, R2, R3, S1, S2, S3.
coupledtempload_s_std_cpe3t.inp	Same as coupledtempload_std_cpe3t.inp except with surface-based loads.

CPE4HT element load tests:

ece4thdd.inp	BX, BY, BF, GRAV, CENT, CENTRIF, CORIO, F1, F2, F3, F4, P1, P2, P3, P4, HP1, HP2, HP3, HP4, R1, R2, R3, R4, S1, S2, S3, S4.
ece4thdr.inp	ROTA.
ece4thdk.inp	HP, P, F, R, S.

CPE4T element load tests:

ece4tfdd.inp	BX, BY, BF, GRAV, CENT, CENTRIF, CORIO, F1, F2, F3, F4, P1, P2, P3, P4, HP1, HP2, HP3, HP4, R1, R2, R3, R4, S1, S2, S3, S4.
ece4tfdr.inp	ROTA.
ece4tfdk.inp	HP, P, F, R, S.

CPE8HT element load tests:

ece8thdd.inp	BX, BY, BF, GRAV, CENT, CENTRIF, CORIO, F1, F2, F3, F4, P1, P2, P3, P4, HP1, HP2, HP3, HP4, R1, R2, R3, R4, S1, S2, S3, S4.
ece8thdr.inp	ROTA.
ece8thdk.inp	HP, P, F, R, S.

CPE8RHT element load tests:

ece8tydd.inp	BX, BY, BF, GRAV, CENT, CENTRIF, CORIO, F1, F2, F3, F4, P1, P2, P3, P4, HP1, HP2, HP3, HP4, R1, R2, R3, R4, S1, S2, S3, S4.
ece8tydr.inp	ROTA.
ece8tydk.inp	HP, P, F, R, S.

CPE8RT element load tests:

ece8trdd.inp	BX, BY, BF, GRAV, CENT, CENTRIF, CORIO, F1, F2, F3, F4, P1, P2, P3, P4, HP1, HP2, HP3, HP4, R1, R2, R3, R4, S1, S2, S3, S4.
ece8trdr.inp	ROTA.
ece8trdk.inp	HP, P, F, R, S.

CPE8T element load tests:

ece8tfdd.inp	BX, BY, BF, GRAV, CENT, CENTRIF, CORIO, F1, F2, F3, F4, P1, P2, P3, P4, HP1, HP2, HP3, HP4, R1, R2, R3, R4, S1, S2, S3, S4.
ece8tfdr.inp	ROTA.
ece8tfdk.inp	HP, P, F, R, S.

CPEG3HT element load tests:

ecg3thdd.inp	BX, BY, BF, GRAV, CENT, CENTRIF, CORIO, F1, P1, HP1, R1, S1.
ecg3thdr.inp	ROTA.
ecg3thdk.inp	HP, P, F, R, S.

CPEG3T element load tests:

ecg3tfdd.inp	BX, BY, BF, GRAV, CENT, CENTRIF, CORIO, F1, P1, HP1, R1, S1.
ecg3tfdr.inp	ROTA.
ecg3tfdk.inp	HP, P, F, R, S.

CPEG4HT element load tests:

ecg4thdd.inp	BX, BY, BF, GRAV, CENT, CENTRIF, CORIO, F1, F2, F3, F4, P1, P2, P3, P4, HP1, HP2, HP3, HP4, R1, R2, R3, R4, S1, S2, S3, S4.
ecg4thdr.inp	ROTA.
ecg4thdk.inp	HP, P, F, R, S.

CPEG4RHT element load tests:

ecg4tydd.inp	BX, BY, BF, GRAV, CENT, CENTRIF, CORIO, F1, F2, F3, F4, P1, P2, P3, P4, HP1, HP2, HP3, HP4, R1, R2, R3, R4, S1, S2, S3, S4.
ecg4tydr.inp	ROTA.
ecg4tydk.inp	HP, P, F, R, S.

CPEG4RT element load tests:

ecg4trdd.inp	BX, BY, BF, GRAV, CENT, CENTRIF, CORIO, F1, F2, F3, F4, P1, P2, P3, P4, HP1, HP2, HP3, HP4, R1, R2, R3, R4, S1, S2, S3, S4.
ecg4trdr.inp	ROTA.
ecg4trdk.inp	HP, P, F, R, S.

CPEG4T element load tests:

ecg4tfdd.inp	BX, BY, BF, GRAV, CENT, CENTRIF, CORIO, F1, F2, F3, F4, P1, P2, P3, P4, HP1, HP2, HP3, HP4, R1, R2, R3, R4, S1, S2, S3, S4.
ecg4tfdr.inp	ROTA.
ecg4tfdk.inp	HP, P, F, R, S.

CPEG6MHT element load tests:

ecg6tldd.inp	BX, BY, BF, GRAV, CENT, CENTRIF, F1, F2, F3, P1, P2, P3, HP1, HP2, HP3,R1, R2, R3, S1, S2, S3.
ecg6tldd_po.inp	**POST OUTPUT* analysis.
ecg6tldr.inp	ROTA.
ecg6tldk.inp	HP, P, F, R, S.

CPEG6MT element load tests:

ecg6tkdd.inp	BX, BY, BF, GRAV, CENT, CENTRIF, F1, F2, F3, P1, P2, P3, HP1, HP2, HP3,R1, R2, R3, S1, S2, S3.
ecg6tkdr.inp	ROTA.
ecg6tkdk.inp	HP, P, F, R, S.

CPEG8HT element load tests:

ecg8thdd.inp	BX, BY, BF, GRAV, CENT, CENTRIF, CORIO, F1, F2, F3, F4, P1, P2, P3, P4, HP1, HP2, HP3, HP4, R1, R2, R3, R4, S1, S2, S3, S4.
ecg8thdr.inp	ROTA.
ecg8thdk.inp	HP, P, F, R, S.

CPEG8RHT element load tests:

ecg8tydd.inp	BX, BY, BF, GRAV, CENT, CENTRIF, CORIO, F1, F2, F3, F4, P1, P2, P3, P4, HP1, HP2, HP3, HP4, R1, R2, R3, R4, S1, S2, S3, S4.
ecg8tydr.inp	ROTA.
ecg8tydk.inp	HP, P, F, R, S.

CPEG8T element load tests:

ecg8tfdd.inp	BX, BY, BF, GRAV, CENT, CENTRIF, CORIO, F1, F2, F3, F4, P1, P2, P3, P4, HP1, HP2, HP3, HP4, R1, R2, R3, R4, S1, S2, S3, S4.
ecg8tfdr.inp	ROTA.
ecg8tfdk.inp	HP, P, F, R, S.

CPS3T element load tests:

coupledtempload_std_cps3t.inp	BX, BY, BF, GRAV, F1, F2, F3, P1, P2, P3, R1, R2, R3, S1, S2, S3.
coupledtempload_s_std_cps3t.inp	Same as coupledtempload_std_cps3t.inp except with surface-based loads.

CPS4T element load tests:

ecs4tfdd.inp	BX, BY, BF, GRAV, CENT, CENTRIF, CORIO, F1, F2, F3, F4, P1, P2, P3, P4, HP1, HP2, HP3, HP4, R1, R2, R3, R4, S1, S2, S3, S4.
ecs4tfdr.inp	ROTA.
ecs4tfdk.inp	HP, P, F, R, S.

CPS8RT element load tests:

ecs8trdd.inp	BX, BY, BF, GRAV, CENT, CENTRIF, CORIO, F1, F2, F3, F4, P1, P2, P3, P4, HP1, HP2, HP3, HP4, R1, R2, R3, R4, S1, S2, S3, S4.
ecs8trdr.inp	ROTA.
ecs8trdk.inp	HP, P, F, R, S.

CPS8T element load tests:

ecs8tfdd.inp	BX, BY, BF, GRAV, CENT, CENTRIF, CORIO, F1, F2, F3, F4, P1, P2, P3, P4, HP1, HP2, HP3, HP4, R1, R2, R3, R4, S1, S2, S3, S4.
ecs8tfdr.inp	ROTA.
ecs8tfdk.inp	HP, P, F, R, S.

Abaqus/Explicit input files

CPE3T element load tests:

coupledtempload_xpl_cpe3t.inp	BX, BY, BF, GRAV, F1, F2, F3, P1, P2, P3, R1, R2, R3, S1, S2, S3.
coupledtempload_s_xpl_cpe3t.inp	Same as coupledtempload_xpl_cpe3t.inp except with surface-based loads.

CPE4RT element load tests:

coupledtemperload_xpl_cpe4rt.inp	BX, BY, BF, GRAV, F1, F2, F3, F4, P1, P2, P3, P4, R1, R2, R3, R4, S1, S2, S3, S4.
coupledtemperload_s_xpl_cpe4rt.inp	Same as coupledtemperload_xpl_cpe4rt.inp except with surface-based loads.

CPE6MT element load tests:

coupledtemperload_xpl_cpe6mt.inp	BX, BY, BF, GRAV, F1, F2, F3, P1, P2, P3, R1, R2, R3, S1, S2, S3.
coupledtemperload_s_xpl_cpe6mt.inp	Same as coupledtemperload_xpl_cpe6mt.inp except with surface-based loads.

CPS3T element load tests:

coupledtemperload_xpl_cps3t.inp	BX, BY, BF, GRAV, F1, F2, F3, P1, P2, P3, R1, R2, R3, S1, S2, S3.
coupledtemperload_s_xpl_cps3t.inp	Same as coupledtemperload_xpl_cps3t.inp except with surface-based loads.

CPS4RT element load tests:

coupledtemperload_xpl_cps4rt.inp	BX, BY, BF, GRAV, F1, F2, F3, F4, P1, P2, P3, P4, R1, R2, R3, R4, S1, S2, S3, S4.
coupledtemperload_s_xpl_cps4rt.inp	Same as coupledtemperload_xpl_cps4rt.inp except with surface-based loads.

CPS6MT element load tests:

coupledtemperload_xpl_cps6mt.inp	BX, BY, BF, GRAV, F1, F2, F3, P1, P2, P3, R1, R2, R3, S1, S2, S3.
coupledtemperload_s_xpl_cps6mt.inp	Same as coupledtemperload_xpl_cps6mt.inp except with surface-based loads.

Coupled temperature-displacement axisymmetric solid elements

Problem description

Model:

Planar dimensions	7×7
Inside radius	1.0
Centrifugal axis of rotation	(0, 1, 0) through origin
Gravity load vector	(0, –1, 0)

Mesh:

Linear elements	2 elements in radial direction
Quadratic elements	1 element in radial direction

Material:

Young's modulus	30×10^6
Poisson's ratio	0.3
Coefficient of thermal expansion	0.0
Thermal conductivity	3.77×10^{-5}
Density	82.9
Sink (bulk fluid) temperature	75.0
Absolute zero temperature	–460.0

Initial conditions:

Hydrostatic pressure datum	3.0
Hydrostatic pressure elevation	0.0

Results and discussion

The calculated reactions are in agreement with the applied loads.

Input files

Abaqus/Standard input files

CAX3T element load tests:

coupledtempload_std_cax3t.inp	BR, BZ, BF, GRAV, F1, F2, F3, P1, P2, P3, R1, R2, R3, S1, S2, S3.
coupledtempload_s_std_cax3t.inp	Same as coupledtempload_std_cax3t.inp except with surface-based loads.

CAX4HT element load tests:

eca4thdd.inp	BR, BZ, BF, GRAV, CENT, CENTRIF, F1, F2, F3, F4, P1, P2, P3, P4, HP1, HP2, HP3, HP4, R1, R2, R3, R4, S1, S2, S3, S4.
eca4thdk.inp	HP, P, F, R, S.

CAX4T element load tests:

eca4tfdd.inp	BR, BZ, BF, GRAV, CENT, CENTRIF, F1, F2, F3, F4, P1, P2, P3, P4, HP1, HP2, HP3, HP4, R1, R2, R3, R4, S1, S2, S3, S4.
eca4tfdk.inp	HP, P, F, R, S.

CAX8HT element load tests:

eca8thdd.inp	BR, BZ, BF, GRAV, CENT, CENTRIF, F1, F2, F3, F4, P1, P2, P3, P4, HP1, HP2, HP3, HP4, R1, R2, R3, R4, S1, S2, S3, S4.
eca8thdk.inp	HP, P, F, R, S.

CAX8RHT element load tests:

eca8tydd.inp	BR, BZ, BF, GRAV, CENT, CENTRIF, F1, F2, F3, F4, P1, P2, P3, P4, HP1, HP2, HP3, HP4, R1, R2, R3, R4, S1, S2, S3, S4.
eca8tydk.inp	HP, P, F, R, S.

CAX8RT element load tests:

eca8trdd.inp	BR, BZ, BF, GRAV, CENT, CENTRIF, F1, F2, F3, F4, P1, P2, P3, P4, HP1, HP2, HP3, HP4, R1, R2, R3, R4, S1, S2, S3, S4.
eca8trdk.inp	HP, P, F, R, S.

CAX8T element load tests:

eca8tfdd.inp	BR, BZ, BF, GRAV, CENT, CENTRIF, F1, F2, F3, F4, P1, P2, P3, P4, HP1, HP2, HP3, HP4, R1, R2, R3, R4, S1, S2, S3, S4.
eca8tfdk.inp	HP, P, F, R, S.

Abaqus/Explicit input files

CAX3T element load tests:

coupledtempload_xpl_cax3t.inp	BR, BZ, BF, GRAV, F1, F2, F3, P1, P2, P3, R1, R2, R3, S1, S2, S3.
coupledtempload_s_xpl_cax3t.inp	Same as coupledtempload_xpl_cax3t.inp except with surface-based loads.

CAX4RT element load tests:

coupledtempload_xpl_cax4rt.inp	BR, BZ, BF, GRAV, F1, F2, F3, F4, P1, P2, P3, P4, R1, R2, R3, R4, S1, S2, S3, S4.
coupledtempload_s_xpl_cax4rt.inp	Same as coupledtempload_xpl_cax4rt.inp except with surface-based loads.

CAX6MT element load tests:

coupledtempload_xpl_cax6mt.inp	BR, BZ, BF, GRAV, F1, F2, F3, P1, P2, P3, R1, R2, R3, S1, S2, S3.
coupledtempload_s_xpl_cax6mt.inp	Same as coupledtempload_xpl_cax6mt.inp except with surface-based loads.

Coupled temperature-displacement three-dimensional elements

Problem description

Model:

Cubic dimensions	$7 \times 7 \times 7$
Centrifugal axis of rotation	(0, 1, 0) through (−1000, 3.5, −3.5)
Gravity load vector	(1, 2, 3)

Material:

Young's modulus	30×10^6
Poisson's ratio	0.3
Coefficient of thermal expansion	0.0
Thermal conductivity	3.77×10^{-5}
Density	82.9
Sink (bulk fluid) temperature	75.0
Absolute zero temperature	−460.0

Initial conditions:

Hydrostatic pressure datum	0.0
Hydrostatic pressure elevation	–7.0

Results and discussion

The calculated reactions are in agreement with the applied loads.

Input files

Abaqus/Standard input files

C3D4T element load tests:

coupledtemploadstdc3d4t.inp	BX, BY, BZ, BF, GRAV, F1, F2, F3, F4, P1, P2, P3, P4, R1, R2, R3, R4, S1, S2, S3, S4.
coupledtemploadsstdc3d4t.inp	Same as coupledtemploadstdc3d4t.inp except with surface-based loads.

C3D6T element load tests:

coupledtemploadstdc3d6t.inp	BX, BY, BZ, BF, GRAV, F1, F2, F3, F4, F5, P1, P2, P3, P4, P5, R1, R2, R3, R4, R5, S1, S2, S3, S4, S5.
coupledtemploadsstdc3d6t.inp	Same as coupledtemploadstdc3d6t.inp except with surface-based loads.

C3D8T element load tests:

ec38tfdd.inp	BX, BY, BZ, BF, GRAV, CENT, CENTRIF, F1, F2, F3, F4, F5, F6, P1, P2, P3, P4, P5, P6, HP1, HP2, HP3, HP4, HP5, HP6, R1, R2, R3, R4, R5, R6,S1, S2, S3, S4, S5, S6.
ec38tfdr.inp	ROTA.
ec38tfdk.inp	HP, P, F, R, S.

C3D8IT element load tests:

coupledtemploadstdc3d8it.inp	BX, BY, BZ, BF, GRAV, CENT, CENTRIF, F1, F2, F3, F4, F5, F6, P1, P2, P3, P4, P5, P6, HP1, HP2, HP3, HP4, HP5, HP6, R1, R2, R3, R4, R5, R6,S1, S2, S3, S4, S5, S6.
coupledtemploadsstdc3d8it.inp	Same as coupledtemploadstdc3d8it.inp except with surface-based loads.

C3D8IHT element load tests:

coupledtemploadstdc3d8iht.inp	BX, BY, BZ, BF, GRAV, CENT, CENTRIF, F1, F2, F3, F4, F5, F6, P1, P2, P3, P4, P5, P6, HP1, HP2, HP3, HP4, HP5, HP6, R1, R2, R3, R4, R5, R6,S1, S2, S3, S4, S5, S6.
coupledtemploadsstdc3d8iht.inp	Same as coupledtemploadstdc3d8iht.inp except with surface-based loads.

C3D8HT element load tests:

ec38thdd.inp	BX, BY, BZ, BF, GRAV, CENT, CENTRIF, F1, F2, F3, F4, F5, F6, P1, P2, P3, P4, P5, P6, HP1, HP2, HP3, HP4, HP5, HP6, R1, R2, R3, R4, R5, R6,S1, S2, S3, S4, S5, S6.
ec38thdd_po.inp	**POST OUTPUT* analysis.
ec38thdr.inp	ROTA.
ec38thdk.inp	HP, P, F, R, S.

C3D20T element load tests:

ec3ktfdd.inp	BX, BY, BZ, BF, GRAV, CENT, CENTRIF, F1, F2, F3, F4, F5, F6, P1, P2, P3, P4, P5, P6, HP1, HP2, HP3, HP4, HP5, HP6, R1, R2, R3, R4, R5, R6,S1, S2, S3, S4, S5, S6.
ec3ktfdr.inp	ROTA.
ec3ktfdk.inp	HP, P, F, R, S.

C3D20HT element load tests:

ec3kthdd.inp	BX, BY, BZ, BF, GRAV, CENT, CENTRIF, F1, F2, F3, F4, F5, F6, P1, P2, P3, P4, P5, P6, HP1, HP2, HP3, HP4, HP5, HP6, R1, R2, R3, R4, R5, R6,S1, S2, S3, S4, S5, S6.
ec3kthdr.inp	ROTA.
ec3kthdk.inp	HP, P, F, R, S.

C3D20RT element load tests:

ec3ktrdd.inp	BX, BY, BZ, BF, GRAV, CENT, CENTRIF, F1, F2, F3, F4, F5, F6, P1, P2, P3, P4, P5, P6, HP1, HP2, HP3, HP4, HP5, HP6, R1, R2, R3, R4, R5, R6,S1, S2, S3, S4, S5, S6.
ec3ktrdr.inp	ROTA.
ec3ktrdk.inp	HP, P, F, R, S.

C3D20RHT element load tests:

ec3ktydd.inp	BX, BY, BZ, BF, GRAV, CENT, CENTRIF, F1, F2, F3, F4, F5, F6, P1, P2, P3, P4, P5, P6, HP1, HP2, HP3, HP4, HP5, HP6, R1, R2, R3, R4, R5, R6,S1, S2, S3, S4, S5, S6.
ec3ktydr.inp	ROTA.
ec3ktydk.inp	HP, P, F, R, S.

Abaqus/Explicit input files

C3D4T element load tests:

coupledtempload_xpl_c3d4t.inp	BX, BY, BZ, BF, GRAV, F1, F2, F3, F4, P1, P2, P3, P4, R1, R2, R3, R4, S1, S2, S3, S4.
coupledtempload_s_xpl_c3d4t.inp	Same as coupledtempload_xpl_c3d4t.inp except with surface-based loads.

C3D6T element load tests:

coupledtempload_xpl_c3d6t.inp	BX, BY, BZ, BF, GRAV, F1, F2, F3, F4, F5, P1, P2, P3, P4, P5, R1, R2, R3, R4, R5, S1, S2, S3, S4, S5.
coupledtempload_s_xpl_c3d6t.inp	Same as coupledtempload_xpl_c3d6t.inp except with surface-based loads.

C3D8RT element load tests:

coupledtempload_xpl_c3d8rt.inp	BX, BY, BZ, BF, GRAV, F1, F2, F3, F4, F5, F6, P1, P2, P3, P4, P5, P6, R1, R2, R3, R4, R5, R6, S1, S2, S3, S4, S5, S6.
coupledtempload_s_xpl_c3d8rt.inp	Same as coupledtempload_xpl_c3d8rt.inp except with surface-based loads.
coupledtempload_s_xpl_sc8rt.inp	Same as coupledtempload_xpl_sc8rt.inp except with surface-based loads.

C3D8T element load test:

coupledtempload_xpl_c3d8t.inp	BX, BY, BZ, BF, GRAV, F1, F2, F3, F4, F5, F6, P1, P2, P3, P4, P5, P6, R1, R2, R3, R4, R5, R6, S1, S2, S3, S4, S5, S6.

C3D8IT element load test:

coupledtempload_xpl_c3d8it.inp	BX, BY, BZ, BF, GRAV, F1, F2, F3, F4, F5, F6, P1, P2, P3, P4, P5, P6, R1, R2, R3, R4, R5, R6, S1, S2, S3, S4, S5, S6.

C3D10T element load tests:

coupledtempload_xpl_c3d10t.inp	BX, BY, BZ, BF, GRAV, F1, F2, F3, F4, P1, P2, P3, P4, R1, R2, R3, R4, S1, S2, S3, S4.
coupledtempload_s_xpl_c3d10t.inp	Same as `coupledtempload_xpl_c3d10t.inp` except with surface-based loads.

C3D10MT element load tests:

coupledtempload_xpl_c3d10mt.inp	BX, BY, BZ, BF, GRAV, F1, F2, F3, F4, P1, P2, P3, P4, R1, R2, R3, R4, S1, S2, S3, S4.
coupledtempload_s_xpl_c3d10mt.inp	Same as `coupledtempload_xpl_c3d10mt.inp` except with surface-based loads.

Coupled temperature-displacement axisymmetric shell element

Problem description

Model:

Length	10.0
Radius	5.0
Thickness	0.5
Centrifugal axis of rotation	(0, 1, 0) through origin
Gravity load vector	(0, 1, 0)

Material:

Young's modulus	3×10^{6}
Poisson's ratio	0.3
Thermal conductivity	3.77×10^{-5}
Density	82.9
Sink (bulk fluid) temperature	75.0
Absolute zero temperature	–460.0

Initial conditions:

Hydrostatic pressure datum	7.0
Hydrostatic pressure elevation	0.0

Results and discussion

The calculated reactions are in agreement with the applied loads.

Input files

SAX2T element load tests:

esa3txdd.inp	BR, BZ, BF, GRAV, CENT, CENTRIF, FNEG, FPOS, P, HP, RNEG, RPOS, SNEG, SPOS.
esa3txdk.inp	HP, P, F, R, S.

Coupled temperature-displacement general shell element

Problem description

Model:

Planar dimensions	7×7
Thickness	2.0
Centrifugal and Coriolis axes of rotation	(0, 1, 0) through origin
Gravity load vector	(0, 1, 0)

Material:

Young's modulus	3×10^{6}
Poisson's ratio	0.3
Thermal conductivity	3.77×10^{-5}
Density	82.9
Sink (bulk fluid) temperature	75.0
Absolute zero temperature	–460.0

Initial conditions:

Hydrostatic pressure datum	7.0
Hydrostatic pressure elevation	0.0

Results and discussion

The calculated reactions are in agreement with the applied loads.

Input files

S4T element load tests:

es34txdd.inp	BX, BY, BZ, BF, GRAV, CENT, CENTRIF, FNEG, FPOS, P, HP, RNEG, RPOS, SNEG, SPOS.
es34txdr.inp	ROTA.
es34txdk.inp	HP, P, F, R, S.

S4RT element load tests:

es4rtxdd.inp	BX, BY, BZ, BF, GRAV, CENT, CENTRIF, FNEG, FPOS, P, HP, RNEG, RPOS, SNEG, SPOS.
es4rtxdr.inp	ROTA.
es4rtxdk.inp	HP, P, F, R, S.

S8RT element load tests:

es38txdd.inp	BX, BY, BZ, BF, GRAV, CENT, CENTRIF, FNEG, FPOS, P, HP, RNEG, RPOS, SNEG, SPOS.
es38txdr.inp	ROTA.
es38txdk.inp	HP, P, F, R, S.

Coupled temperature-displacement axisymmetric twist elements

Problem description

Model:

Planar dimensions	7×7
Inside radius	1.0
Centrifugal axis of rotation	(0, 1, 0) through origin
Gravity load vector	(0, −1, 0)

Mesh:

Linear elements	2 elements in radial direction
Quadratic elements	1 element in radial direction

Material:

Young's modulus	30×10^6
Poisson's ratio	0.3
Coefficient of thermal expansion	0.0
Thermal conductivity	3.77×10^{-5}
Density	82.9
Sink (bulk fluid) temperature	75.0
Absolute zero temperature	−460.0

Initial conditions:

Hydrostatic pressure datum	3.0
Hydrostatic pressure elevation	0.0

Results and discussion

The calculated reactions are in agreement with the applied loads.

Input files

CGAX3T element load tests:

eca3hfdd.inp	BR, BZ, BF, CENT, CENTRIF, GRAV, F1, F2, F3, P1, P2, P3, HP1, HP2, HP3, R1, R2, R3, S1, S2, S3.
eca3hfdk.inp	HP, P, F, R, S.

CGAX3HT element load tests:

eca3hhdd.inp	BR, BZ, BF, CENT, CENTRIF, GRAV, F1, F2, F3, P1, P2, P3, HP1, HP2, HP3, R1, R2, R3, S1, S2, S3.
eca3hhdk.inp	HP, P, F, R, S.

CGAX4T element load tests:

eca4hfdd.inp	BR, BZ, BF, CENT, CENTRIF, GRAV, F1, F2, F3, F4, P1, P2, P3, P4, HP1, HP2, HP3, HP4, R1, R2, R3, R4, S1, S2, S3, S4.
eca4hfdk.inp	HP, P, F, R, S.

CGAX4HT element load tests:

eca4hhdd.inp	BR, BZ, BF, CENT, CENTRIF, GRAV, F1, F2, F3, F4, P1, P2, P3, P4, HP1, HP2, HP3, HP4, R1, R2, R3, R4, S1, S2, S3, S4.
eca4hhdk.inp	HP, P, F, R, S.

CGAX4RT element load tests:

eca4hrdd.inp	BR, BZ, BF, CENT, CENTRIF, GRAV, F1, F2, F3, F4, P1, P2, P3, P4, HP1, HP2, HP3, HP4, R1, R2, R3, R4, S1, S2, S3, S4.
eca4hrdk.inp	HP, P, F, R, S.

CGAX4RHT element load tests:

eca4hydd.inp	BR, BZ, BF, CENT, CENTRIF, GRAV, F1, F2, F3, F4, P1, P2, P3, P4, HP1, HP2, HP3, HP4, R1, R2, R3, R4, S1, S2, S3, S4.
eca4hydk.inp	HP, P, F, R, S.

CGAX6MT element load tests:

eca6hfdd.inp	BR, BZ, BF, CENT, CENTRIF, GRAV, F1, F2, P1, P2, HP1, HP2, R1, R2, S1, S2.
eca6hfdk.inp	HP, P, F, R, S.

CGAX6MHT element load tests:

eca6hhdd.inp	BR, BZ, BF, CENT, CENTRIF, GRAV, F1, F2, P1, P2, HP1, HP2, R1, R2, S1, S2.

eca6hhdk.inp	HP, P, F, R, S.
CGAX8T element load tests:	
eca8hfdd.inp	BR, BZ, BF, CENT, CENTRIF, GRAV, F1, F2, F3, F4, P1, P2, P3, P4, HP1, HP2, HP3, HP4, R1, R2, R3, R4, S1, S2, S3, S4.
eca8hfdk.inp	HP, P, F, R, S.
CGAX8HT element load tests:	
eca8hhdd.inp	BR, BZ, BF, CENT, CENTRIF, GRAV, F1, F2, F3, F4, P1, P2, P3, P4, HP1, HP2, HP3, HP4, R1, R2, R3, R4, S1, S2, S3, S4.
eca8hhdk.inp	HP, P, F, R, S.
CGAX8RT element load tests:	
eca8hrdd.inp	BR, BZ, BF, CENT, CENTRIF, GRAV, F1, F2, F3, F4, P1, P2, P3, P4, HP1, HP2, HP3, HP4, R1, R2, R3, R4, S1, S2, S3, S4.
eca8hrdk.inp	HP, P, F, R, S.
CGAX8RHT element load tests:	
eca8hydd.inp	BR, BZ, BF, CENT, CENTRIF, GRAV, F1, F2, F3, F4, P1, P2, P3, P4, HP1, HP2, HP3, HP4, R1, R2, R3, R4, S1, S2, S3, S4.
eca8hydk.inp	HP, P, F, R, S.

Coupled temperature-displacement continuum shell elements

Problem description

Model:

Cubic dimensions	$7 \times 7 \times 7$
Centrifugal axis of rotation	(0, 1, 0) through (−1000, 3.5, −3.5)
Gravity load vector	(1, 2, 3)

Material:

Young's modulus	30×10^{6}
Poisson's ratio	0.3
Coefficient of thermal expansion	0.0
Thermal conductivity	3.77×10^{-5}
Density	82.9
Sink (bulk fluid) temperature	75.0
Absolute zero temperature	−460.0

Initial conditions:

Hydrostatic pressure datum	0.0
Hydrostatic pressure elevation	−7.0

Results and discussion

The calculated reactions are in agreement with the applied loads.

Input files

Abaqus/Standard input files

SC8RT element load tests:

esc8tfdd.inp	BX, BY, BZ, BF, GRAV, CENT, CENTRIF, F1, F2, F3, F4, F5, F6, P1, P2, P3, P4, P5, P6, HP1, HP2, HP3, HP4, HP5, HP6, R1, R2, R3, R4, R5, R6,S1, S2, S3, S4, S5, S6.
esc8tsdd.inp	Same as esc8tfdd.inp except with surface-based loads.
esc8tfdr.inp	ROTA.
esc8tfdk.inp	HP, P, F, R, S.

Abaqus/Explicit input file

coupledtemperload_xpl_sc8rt.inp	BX, BY, BZ, BF, GRAV, F1, F2, F3, F4, F5, F6, P1, P2, P3, P4, P5, P6, R1, R2, R3, R4, R5, R6, S1, S2, S3, S4, S5, S6.

SC8RT element load test:

Coupled thermal-electrical-structural elements

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Coupled thermal-electrical-structural axisymmetric solid elements

Problem description

Model:

Planar dimensions	7×7
Inside radius	1.0
Centrifugal axis of rotation	(0, 1, 0) through origin
Gravity load vector	(0, −1, 0)

Material:

Young's modulus	30×10^{6}
Poisson's ratio	0.3
Coefficient of thermal expansion	0.0
Thermal conductivity	3.77×10^{-5}
Electrical conductivity	3.77×10^{-5}
Joule heat fraction	0.0
Density	82.9
Sink (bulk fluid) temperature	75.0
Absolute zero temperature	−460.0

Initial conditions:

Hydrostatic pressure datum	3.0
Hydrostatic pressure elevation	0.0

Results and discussion

The calculated reactions are in agreement with the applied loads.

Input files

Abaqus/Standard input files

QAX3 element load tests:

tes_qax3.inp	BR, BZ, BF, CBF, CS1, CS2, CS3, GRAV, F1, F2, F3, P1, P2, P3, R1, R2, R3, S1, S2, S3.
tes_qax3_sload.inp	Same as tes_qax3.inp except with surface-based loads.
QAX4 element load tests:	
tes_qax4.inp	BR, BZ, BF, CBF, CS1, CS2, CS3, CS4, GRAV, CENTRIF, F1, F2, F3, F4, P1, P2, P3, P4, HP1, HP2, HP3, HP4, R1, R2, R3, R4, S1, S2, S3, S4.
tes_qax4_sload.inp	S, CS, HP, P, F, R.
QAX8 element load tests:	
tes_qax8.inp	BR, BZ, BF, CBF, CS1, CS2, CS3, CS4, GRAV, CENTRIF, F1, F2, F3, F4, P1, P2, P3, P4, HP1, HP2, HP3, HP4, R1, R2, R3, R4, S1, S2, S3, S4.
tes_qax8_sload.inp	S, CS, HP, P, F, R.

Coupled thermal-electrical-structural three-dimensional elements

Problem description

Model:

Cubic dimensions	$7 \times 7 \times 7$
Centrifugal axis of rotation	(0, 1, 0) through (−1000, 3.5, −3.5)
Gravity load vector	(1, 2, 3)

Material:

Young's modulus	30×10^{6}
Poisson's ratio	0.3
Coefficient of thermal expansion	0.0
Thermal conductivity	3.77×10^{-5}
Electrical conductivity	3.77×10^{-5}
Joule heat fraction	0.0
Density	82.9
Sink (bulk fluid) temperature	75.0
Absolute zero temperature	−460.0

Initial conditions:

Hydrostatic pressure datum	0.0
Hydrostatic pressure elevation	−7.0

Results and discussion

The calculated reactions are in agreement with the applied loads.

Input files

Abaqus/Standard input files

Q3D4 element load tests:

tes_q3d4.inp	BX, BY, BZ, BF, CBF, CS1, CS2, CS3, CS4, GRAV, F1, F2, F3, F4, P1, P2, P3, P4, R1, R2, R3, R4, S1, S2, S3, S4.
tes_q3d4_sload.inp	Same as tes_q3d4.inp except with surface-based loads.

Q3D6 element load tests:

tes_q3d6.inp	BX, BY, BZ, BF, CBF, CS1, CS2, CS3, CS4, CS5, GRAV, F1, F2, F3, F4, F5, P1, P2, P3, P4, P5, R1, R2, R3, R4, R5, S1, S2, S3, S4, S5.
tes_q3d6_sload.inp	Same as tes_q3d6.inp except with surface-based loads.

Q3D8 element load tests:

tes_q3d8.inp	BX, BY, BZ, BF, CBF, CS1, CS2, CS3, CS4, CS5, CS6, GRAV, CENT, CENTRIF, F1, F2, F3, F4, F5, F6, P1, P2, P3, P4, P5, P6, HP1, HP2, HP3, HP4, HP5, HP6, R1, R2, R3, R4, R5, R6,S1, S2, S3, S4, S5, S6.
tes_q3d8_rota.inp	ROTA.
tes_q3d8_sload.inp	CS, HP, P, F, R, S.

Q3D8H element load tests:

tes_q3d8h.inp	BX, BY, BZ, BF, CBF, CS1, CS2, CS3, CS4, CS5, CS6, GRAV, CENT, CENTRIF, F1, F2, F3, F4, F5, F6, P1, P2, P3, P4, P5, P6, HP1, HP2, HP3, HP4, HP5, HP6, R1, R2, R3, R4, R5, R6,S1, S2, S3, S4, S5, S6.
tes_q3d8h_po.inp	**POST OUTPUT* analysis.
tes_q3d8h_rota.inp	ROTA.
tes_q3d8h_sload.inp	CS, HP, P, F, R, S.

Q3D8R element load tests:

tes_q3d8r.inp	BX, BY, BZ, BF, CBF, CS1, CS2, CS3, CS4, CS5, CS6, GRAV, CENT, CENTRIF, F1, F2, F3, F4, F5, F6, P1, P2, P3, P4, P5, P6, HP1, HP2, HP3, HP4, HP5, HP6, R1, R2, R3, R4, R5, R6,S1, S2, S3, S4, S5, S6.
tes_q3d8r_rota.inp	ROTA.
tes_q3d8r_sload.inp	CS, HP, P, F, R, S.

Q3D8RH element load tests:

tes_q3d8rh.inp	BX, BY, BZ, BF, CBF, CS1, CS2, CS3, CS4, CS5, CS6, GRAV, CENT, CENTRIF, F1, F2, F3, F4, F5, F6, P1, P2, P3, P4, P5, P6, HP1, HP2, HP3, HP4, HP5, HP6, R1, R2, R3, R4, R5, R6,S1, S2, S3, S4, S5, S6.
tes_q3d8rh_rota.inp	ROTA.
tes_q3d8rh_sload.inp	CS, HP, P, F, R, S.

Q3D10M element load tests:

tes_q3d10m.inp	BX, BY, BZ, BF, CBF, CS1, CS2, CS3, CS4, GRAV, F1, F2, F3, F4, P1, P2, P3, P4, R1, R2, R3, R4, S1, S2, S3, S4.
tes_q3d10m_sload.inp	Same as tes_q3d10m.inp except with surface-based loads.

Q3D10MH element load tests:

tes_q3d10mh.inp	BX, BY, BZ, BF, CBF, CS1, CS2, CS3, CS4, GRAV, F1, F2, F3, F4, P1, P2, P3, P4, R1, R2, R3, R4, S1, S2, S3, S4.
tes_q3d10mh_sload.inp	Same as tes_q3d10mh.inp except with surface-based loads.
Q3D20 element load tests:	
tes_q3d20.inp	BX, BY, BZ, BF, CBF, CS1, CS2, CS3, CS4, CS5, CS6, GRAV, CENT, CENTRIF, F1, F2, F3, F4, F5, F6, P1, P2, P3, P4, P5, P6, HP1, HP2, HP3, HP4, HP5, HP6, R1, R2, R3, R4, R5, R6,S1, S2, S3, S4, S5, S6.
tes_q3d20_rota.inp	ROTA.
tes_q3d20_sload.inp	CS, HP, P, F, R, S.
Q3D20H element load tests:	
tes_q3d20h.inp	BX, BY, BZ, BF, CBF, CS1, CS2, CS3, CS4, CS5, CS6, GRAV, CENT, CENTRIF, F1, F2, F3, F4, F5, F6, P1, P2, P3, P4, P5, P6, HP1, HP2, HP3, HP4, HP5, HP6, R1, R2, R3, R4, R5, R6,S1, S2, S3, S4, S5, S6.
tes_q3d20_rota.inp	ROTA.
tes_q3d20_sload.inp	CS, HP, P, F, R, S.
Q3D20R element load tests:	
tes_q3d20r.inp	BX, BY, BZ, BF, CBF, CS1, CS2, CS3, CS4, CS5, CS6, GRAV, CENT, CENTRIF, F1, F2, F3, F4, F5, F6, P1, P2, P3, P4, P5, P6, HP1, HP2, HP3, HP4, HP5, HP6, R1, R2, R3, R4, R5, R6,S1, S2, S3, S4, S5, S6.
tes_q3d20r_rota.inp	ROTA.
tes_q3d20r_sload.inp	CS, HP, P, F, R, S.
Q3D20RH element load tests:	
tes_q3d20rh.inp	BX, BY, BZ, BF, CBF, CS1, CS2, CS3, CS4, CS5, CS6, GRAV, CENT, CENTRIF, F1, F2, F3, F4, F5, F6, P1, P2, P3, P4, P5, P6, HP1, HP2, HP3, HP4, HP5, HP6, R1, R2, R3, R4, R5, R6,S1, S2, S3, S4, S5, S6.
tes_q3d20rh_rota.inp	ROTA.
tes_q3d20rh_sload.inp	CS, HP, P, F, R, S.

Piezoelectric elements

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Piezoelectric elements have both displacements and electric potentials as degrees of freedom. These elements include truss, plane stress, plane strain, axisymmetric, or three-dimensional continuum. The elements are identical to the basic stress/displacement elements except for the coupling between the stress field and the electrical potential gradients. The mechanical loads are tested for these elements but are not reported here since they are identical to those reported in the section for continuum stress/displacement elements. Only the additional loads associated with body and distributed charges are reported in this section.

Truss elements

Problem description

Model:

Length	1.0
Area	0.1
Centrifugal axis of rotation	(0, 1, 0) through (.5, 0, 0)
Gravitational load vector	(0, −1, 0)

Material:

Young's modulus	3×10^{6}
Coefficient of thermal expansion	.0001
Density	5×10^{-5}
Piezoelectric coupling matrix	
Dielectric term	5.872×10^{-9}

$$\begin{bmatrix} 15.08 & 0 & 0 \\ -5.207 & 0 & 0 \\ -5.207 & 0 & 0 \\ 0 & 12.710 & 0 \\ 0 & 0 & 12.710 \\ 0 & 0 & 0 \end{bmatrix}$$

Initial conditions:

Initial temperature	ALL, −10.0

Results and discussion

The calculated reactions are in agreement with the applied loads.

Input files

T2D2E element load tests:

et22efdf.inp	BX, BY, GRAV, CENT, CENTRIF, EBF.
et22efdr.inp	ROTA.

T3D2E element load tests:

et32efdf.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, EBF.
et32efdr.inp	ROTA.

T2D3E element load tests:

et23efdf.inp	BX, BY, GRAV, CENT, CENTRIF, EBF.
et23efdr.inp	ROTA.

T3D3E element load tests:

et33efdf.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, EBF.
et33efdr.inp	ROTA.

Plane stress and plane strain elements

Problem description

Model:

Square dimensions	7×7
Thickness	1.0
Centrifugal axis of rotation	(0, 1, 0) through origin
Gravitational load vector	(0, −1, 0)

Material:

Young's modulus	3×10^{6}
Poisson's ratio	0.3
Coefficient of thermal expansion	.0001
Density	5×10^{-5}
Piezoelectric coupling matrix	
Dielectric term	5.872×10^{-9}

$$\begin{bmatrix} 15.08 & 0 & 0 \\ -5.207 & 0 & 0 \\ -5.207 & 0 & 0 \\ 0 & 12.710 & 0 \\ 0 & 0 & 12.710 \\ 0 & 0 & 0 \end{bmatrix}$$

Initial conditions:

Initial temperature	ALL, −10.0
Hydrostatic pressure datum	lower-order elements: 7.0

	higher-order elements: 3.0
Hydrostatic pressure elevation	0.0

Results and discussion

The calculated reactions are in agreement with the applied loads.

Input files

CPS3E element load tests:

ecs3efdf.inp	BX, BY, GRAV, CENT, CENTRIF, P1, P2, P3, HP1, HP2, HP3, EBF, ES1, ES2, ES3.
ecs3efdr.inp	ROTA.
ecs3efdm.inp	ES, HP, P.

CPE3E element load tests:

ece3efdf.inp	BX, BY, GRAV, CENT, CENTRIF, P1, P2, P3, HP1, HP2, HP3, EBF, ES1, ES2, ES3.
ece3efdr.inp	ROTA.
ece3efdm.inp	ES, HP, P.

CPS4E element load tests:

ecs4efdf.inp	BX, BY, GRAV, CENT, CENTRIF, P1, P2, P3, P4, HP1, HP2, HP3, HP4, EBF, ES1, ES2, ES3, ES4.
ecs4efdr.inp	ROTA.
ecs4efdm.inp	ES, HP, P.

CPE4E element load tests:

ece4efdf.inp	BX, BY, GRAV, CENT, CENTRIF, P1, P2, P3, P4, HP1, HP2, HP3, HP4, EBF, ES1, ES2, ES3, ES4.
ece4efdr.inp	ROTA.
ece4efdm.inp	ES, HP, P.

CPS6E element load tests:

ecs6efdf.inp	BX, BY, GRAV, CENT, CENTRIF, P1, P2, P3, HP1, HP2, HP3, EBF, ES1, ES2, ES3.
ecs6efdr.inp	ROTA.
ecs6efdm.inp	ES, HP, P.

CPE6E element load tests:

ece6efdf.inp	BX, BY, GRAV, CENT, CENTRIF, P1, P2, P3, HP1, HP2, HP3, EBF, ES1, ES2, ES3.
ece6efdr.inp	ROTA.
ece6efdm.inp	ES, HP, P.

CPS8E element load tests:

ecs8efdf.inp	BX, BY, GRAV, CENT, CENTRIF, P1, P2, P3, P4, HP1, HP2, HP3, HP4, EBF, ES1, ES2, ES3, ES4.

ecs8efdr.inp	ROTA.
ecs8efdm.inp	ES, HP, P.
CPE8E element load tests:	
ece8efdf.inp	BX, BY, GRAV, CENT, CENTRIF, P1, P2, P3, P4, HP1, HP2, HP3, HP4, EBF, ES1, ES2, ES3, ES4.
ece8efdr.inp	ROTA.
ece8efdm.inp	ES, HP, P.
CPS8RE element load tests:	
ecs8erdf.inp	BX, BY, GRAV, CENT, CENTRIF, P1, P2, P3, P4, HP1, HP2, HP3, HP4, EBF, ES1, ES2, ES3, ES4.
ecs8erdr.inp	ROTA.
ecs8erdm.inp	ES, HP, P.
CPE8RE element load tests:	
ece8erdf.inp	BX, BY, GRAV, CENT, CENTRIF, P1, P2, P3, P4, HP1, HP2, HP3, HP4, EBF, ES1, ES2, ES3, ES4.
ece8erdr.inp	ROTA.
ece8erdm.inp	ES, HP, P.

Axisymmetric elements

Problem description

Model:

Planar dimensions	3×3
Inside radius	1.0
Outside radius	4.0
Centrifugal axis of rotation	(0, 1, 0) through origin
Gravitational load vector	(0, −1, 0)

Material:

Young's modulus	3×10^{6}
Poisson's ratio	0.3
Coefficient of thermal expansion	.0001
Density	5×10^{-5}
Piezoelectric coupling matrix	
Dielectric term	5.872×10^{-9}

$$\begin{bmatrix} 15.08 & 0 & 0 \\ -5.207 & 0 & 0 \\ -5.207 & 0 & 0 \\ 0 & 12.710 & 0 \\ 0 & 0 & 12.710 \\ 0 & 0 & 0 \end{bmatrix}$$

Initial conditions:

Hydrostatic pressure datum	3.0
Hydrostatic pressure elevation	0.0

Results and discussion

The calculated reactions are in agreement with the applied loads.

Input files

CAX3E element load tests:

eca3efdf.inp	BZ, GRAV, CENT, P1, P2, P3, HP1, HP2, HP3, EBF, ES1, ES2, ES3.
eca3efdm.inp	ES, HP, P.

CAX4E element load tests:

eca4efdf.inp	BZ, GRAV, CENT, P1, P2, P3, P4, HP1, HP2, HP3, HP4, EBF, ES1, ES2, ES3, ES4.
eca4efdm.inp	ES, HP, P.

CAX6E element load tests:

eca6efdf.inp	BZ, GRAV, CENT, P1, P2, P3, HP1, HP2, HP3, EBF, ES1, ES2, ES3.
eca6efdm.inp	ES, HP, P.

CAX8E element load tests:

eca8efdf.inp	BZ, GRAV, CENT, P1, P2, P3, P4, HP1, HP2, HP3, HP4, EBF, ES1, ES2, ES3, ES4.
eca8efdm.inp	ES, HP, P.

CAX8RE element load tests:

eca8erdf.inp	BZ, GRAV, CENT, P1, P2, P3, P4, HP1, HP2, HP3, HP4, EBF, ES1, ES2, ES3, ES4.
eca8erdm.inp	ES, HP, P.

Three-dimensional solids

Problem description

Model:

Cubic dimensions	$7 \times 7 \times 7$
Centrifugal axes of rotation	(0, 1, 0) through (−1000, 3.5, −3.5)
Gravitational load vector	(1, 0, 0)

Material:

Young's modulus	3×10^6
Poisson's ratio	0.3

Coefficient of thermal expansion	.0001
Density	10.0
Piezoelectric coupling matrix	
Dielectric term	5.872×10^{-9}

$$\begin{bmatrix} 15.08 & 0 & 0 \\ -5.207 & 0 & 0 \\ -5.207 & 0 & 0 \\ 0 & 12.710 & 0 \\ 0 & 0 & 12.710 \\ 0 & 0 & 0 \end{bmatrix}$$

Initial conditions:

Initial temperature	ALL, −10.0
Hydrostatic pressure datum	0.0
Hydrostatic pressure elevation	−7.0

Results and discussion

The calculated reactions are in agreement with the applied loads.

Input files

C3D4E element load tests:

ec34efdf.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, P1, P2, P3, P4, HP1, HP2, HP3, HP4, EBF, ES1, ES2, ES3, ES4.
ec34efdr.inp	ROTA.
ec34efdm.inp	ES, HP, P.

C3D6E element load tests:

ec36efdf.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, P1, P2, P3, P4, P5, HP1, HP2, HP3, HP4, HP5, EBF, ES1, ES2, ES3, ES4, ES5.
ec36efdr.inp	ROTA.
ec36efdm.inp	ES, HP, P.

C3D8E element load tests:

ec38efdf.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, P1, P2, P3, P4, P5, P6, HP1, HP2, HP3, HP4, HP5, HP6, EBF, ES1, ES2, ES3, ES4, ES5, ES6.
ec38efdr.inp	ROTA.
ec38efdm.inp	ES, HP, P.

C3D10E element load tests:

ec3aefdf.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, P1, P2, P3, P4, HP1, HP2, HP3, HP4, EBF, ES1, ES2, ES3, ES4.
ec3aefdr.inp	ROTA.
ec3aefdm.inp	ES, HP, P.

C3D15E element load tests:

ec3fefdf.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, P1, P2, P3, P4, P5, HP1, HP2, HP3, HP4, HP5, EBF, ES1, ES2, ES3, ES4, ES5.
ec3fefdr.inp	ROTA.
ec3fefdm.inp	ES, HP, P.

C3D20E element load tests:

ec3kefdf.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, P1, P2, P3, P4, P5, P6, HP1, HP2, HP3, HP4, HP5, HP6, EBF, ES1, ES2, ES3, ES4, ES5, ES6.
ec3kefdr.inp	ROTA.
ec3kefdm.inp	ES, HP, P.

C3D20RE element load tests:

ec3kerdf.inp	BX, BY, BZ, GRAV, CENT, CENTRIF, P1, P2, P3, P4, P5, P6, HP1, HP2, HP3, HP4, HP5, HP6, EBF, ES1, ES2, ES3, ES4, ES5, ES6.
ec3kerdr.inp	ROTA.
ec3kerdm.inp	ES, HP, P.

Continuum mass diffusion elements

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Planar solid mass diffusion elements

Problem description

Model:

Square dimension	7 × 7
Thickness	1.0

Material:

Solubility	1.0

Results and discussion

The calculated reactions are in agreement with the applied loads.

Input files

ec23mfdc.inp	DC2D3; Diffusivity: 3.77×10^{-5} ($\boldsymbol{D_{11}}$), 7.54×10^{-5} ($\boldsymbol{D_{22}}$), 11.31×10^{-5} ($\boldsymbol{D_{33}}$); Loads: BF, S1, S2, S3.
ec23mfdc.inp	DC2D3; Diffusivity: 3.77×10^{-5} ($\boldsymbol{D_{11}}$), 7.54×10^{-5} ($\boldsymbol{D_{22}}$), 11.31×10^{-5} ($\boldsymbol{D_{33}}$); Loads: BF, S1, S2, S3.
ec24mfdc.inp	DC2D4; Diffusivity: 3.77×10^{-5}; Loads: BF, S1, S2, S3, S4.
ec26mfdc.inp	DC2D6; Diffusivity: 3.77×10^{-5} ($\boldsymbol{D_{11}}$), 3.77×10^{-6} ($\boldsymbol{D_{12}}$), 7.54×10^{-5} ($\boldsymbol{D_{22}}$), 3.77×10^{-6} ($\boldsymbol{D_{13}}$), 3.77×10^{-6} ($\boldsymbol{D_{23}}$), 11.31×10^{-5} ($\boldsymbol{D_{33}}$); Loads: BF, S1, S2, S3.
ec28mfdc.inp	DC2D8; Diffusivity: 3.77×10^{-5}; Loads: BF, S1, S2, S3, S4.

Axisymmetric solid mass diffusion elements

Problem description

Model:

Planar dimensions	7 × 7
Inside radius	1.0

Material:

Diffusivity	3.77×10^{-5}
Solubility	1.0

Results and discussion

The calculated reactions are in agreement with the applied loads.

Input files

eca3mfdc.inp	DCAX3: BF, S1, S2, S3.
eca4mfdc.inp	DCAX4: BF, S1, S2, S3, S4.
eca6mfdc.inp	DCAX6: BF, S1, S2, S3.
eca8mfdc.inp	DCAX8: BF, S1, S2, S3, S4.

Three-dimensional solid mass diffusion elements

Problem description

Model:

Cubic dimensions	$7 \times 7 \times 7$

Material:

Diffusivity	3.77×10^{-5}
Solubility	1.0

Results and discussion

The calculated reactions are in agreement with the applied loads.

Input files

ec34mfdc.inp	DC3D4: BF, S1, S2, S3, S4.
ec36mfdc.inp	DC3D6: BF, S1, S2, S3, S4, S5.
ec36mfdc_po.inp	**POST OUTPUT* analysis.
ec38mfdc.inp	DC3D8: BF, S1, S2, S3, S4, S5, S6.
ec3amfdc.inp	DC3D10: BF, S1, S2, S3, S4.
ec3amfdc_po.inp	**POST OUTPUT* analysis.
ec3fmfdc.inp	DC3D15: BF, S1, S2, S3, S4, S5.
ec3kmfdc.inp	DC3D20: BF, S1, S2, S3, S4, S5, S6.

Thermal-electrical elements

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

One-dimensional thermal-electrical elements

Problem description

Model:

Length	7.0
Area	3.0

Material:

Thermal conductivity	3.77×10^{-5}
Electrical conductivity	3.77×10^{-5}
Joule heat fraction	0.0

Results and discussion

The calculated reactions are in agreement with the applied loads.

Input files

ec12vfdf.inp	DC1D2E: BF, S1, S2, CBF, CS1, CS2.
ec13vfdf.inp	DC1D3E: BF, S1, S2, CBF, CS1, CS2.

Planar thermal-electrical elements

Problem description

Model:

Square dimension	7×7
Thickness	1.0

Material:

Thermal conductivity	3.77×10^{-5}
Electrical conductivity	3.77×10^{-5}
Joule heat fraction	0.0

Results and discussion

The calculated reactions are in agreement with the applied loads.

Input files

DC2D3E element load tests:

ec23vfdf.inp	BF, S1, S2, S3, CBF, CS1, CS2, CS3.
ec23vfdm.inp	S, CS.

DC2D4E element load tests:

ec24vfdf.inp	BF, S1, S2, S3, S4, CBF, CS1, CS2, CS3, CS4.
ec24vfdf_po.inp	**POST OUTPUT* analysis.
ec24vfdm.inp	S, CS.

DC2D6E element load tests:

ec26vfdf.inp	BF, S1, S2, S3, CBF, CS1, CS2, CS3.
ec26vfdm.inp	S, CS.

DC2D8E element load tests:

ec28vfdf.inp	BF, S1, S2, S3, S4, CBF, CS1, CS2, CS3, CS4.
ec28vfdm.inp	S, CS.

Axisymmetric thermal-electrical elements

Problem description

Model:

Planar dimensions	7×7
Inside radius	1.0

Material:

Thermal conductivity	3.77×10^{-5}
Electrical conductivity	3.77×10^{-5}
Joule heat fraction	0.0

Results and discussion

The calculated reactions are in agreement with the applied loads.

Input files

DCAX3E element load tests:

eca3vfdf.inp	BF, S1, S2, S3, CBF, CS1, CS2, CS3.
eca3vfdm.inp	S, CS.

DCAX4E element load tests:

eca4vfdf.inp	BF, S1, S2, S3, S4, CBF, CS1, CS2, CS3, CS4.
eca4vfdm.inp	S, CS.

DCAX6E element load tests:

eca6vfdf.inp	BF, S1, S2, S3, CBF, CS1, CS2, CS3.
eca6vfdm.inp	S, CS.

DCAX8E element load tests:

eca8vfdf.inp	BF, S1, S2, S3, S4, CBF, CS1, CS2, CS3, CS4.
eca8vfdm.inp	S, CS.

Three-dimensional thermal-electrical elements

Problem description

Model:

Cubic dimensions	$7 \times 7 \times 7$

Material:

Thermal conductivity	3.77×10^{-5}
Electrical conductivity	3.77×10^{-5}
Joule heat fraction	0.0

Results and discussion

The calculated reactions are in agreement with the applied loads.

Input files

DC3D4E element load tests:

ec34vfdf.inp	BF, S1, S2, S3, S4, CBF, CS1, CS2, CS3, CS4.
ec34vfdf_po.inp	**POST OUTPUT* analysis.
ec34vfdm.inp	S, CS.

DC3D6E element load tests:

ec36vfdf.inp	BF, S1, S2, S3, S4, S5, CBF, CS1, CS2, CS3, CS4, CS5.
ec36vfdm.inp	S, CS.

DC3D8E element load tests:

ec38vfdf.inp	BF, S1, S2, S3, S4, S5, S6, CBF, CS1, CS2, CS3, CS4, CS5, CS6.
er38vfdf.inp	BF, S1, S2, S3, S4, S5, S6, CBF, CS1, CS2, CS3, CS4, CS5, CS6.
ec38vfdf_po.inp	**POST OUTPUT* analysis.
ec38vfdm.inp	S, CS.

DC3D10E element load tests:

ec3avfdf.inp	BF, S1, S2, S3, S4, CBF, CS1, CS2, CS3, CS4.

ec3avfdm.inp	S, CS.
DC3D15E element load tests:	
ec3fvfdf.inp	BF, S1, S2, S3, S4, S5, CBF, CS1, CS2, CS3, CS4, CS5.
ec3fvfdm.inp	S, CS.
DC3D20E element load tests:	
ec3kvfdf.inp	BF, S1, S2, S3, S4, S5, S6, CBF, CS1, CS2, CS3, CS4, CS5, CS6.
ec3kvfdm.inp	S, CS.

Rigid elements

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Two-dimensional rigid element

Problem description

Model:

Length	1.0
Thickness	0.1
Centrifugal axis of rotation	(1, 0, 0) through (1, 1, 0)

Results and discussion

The calculated reactions are in agreement with the applied loads.

Input files

erp2sxd1.inp	R2D2: CENT, BX, BY.

Axisymmetric rigid element

Problem description

Model:

Length of link	10
Thickness of link	0.5
Radius	5
Centrifugal axis of rotation	(0, 1, 0) through origin

Initial conditions:

Hydrostatic pressure datum	10.0
Hydrostatic pressure elevation	0.0

Results and discussion

The calculated reactions are in agreement with the applied loads.

Input files

era2sxd1.inp	RAX2: CENT, BR, BZ, P, HP.
era2sxdi.inp	RAX2: P, HP.

Three-dimensional rigid elements

Problem description

Model:

Square dimensions	7×7
Thickness	2.0
Centrifugal axis of rotation	(0, 1, 0) through origin

Initial conditions:

Hydrostatic pressure datum	7.0
Hydrostatic pressure elevation	0.0

Results and discussion

The calculated reactions are in agreement with the applied loads.

Input files

er33sxd1.inp	R3D3: CENT, BX, BY, BZ, P, HP.
er33sxdi.inp	R3D3: P, HP.
er34sxd1.inp	R3D4: CENT, BX, BY, BZ, P, HP.
er34sxdi.inp	R3D4: P, HP.

Mass and rotary inertia elements

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Problem description

Model:

Isotropic mass	
M	100.0
Anisotropic mass	
M_x	100.0
M_y	200.0
M_z	300.0
Rotary inertia	
I_{11}	100.0
I_{22}	200.0
I_{33}	300.0
Centrifugal axis of rotation	(0, 0, 1) through (0, 0, 0)
Gravity load vector	(0, 1, 0)
Rotary acceleration axis	(0, 0, 1) through (0, 0, 0)

General: The ROTARYI element and anisotropic MASS element are also tested with local coordinate systems and with finite rotation.

Results and discussion

The calculated reactions are in agreement with the applied loads.

Input files

emassd1.inp	Isotropic MASS: GRAV, CENTRIF, ROTA.
emassd_anis.inp	Anisotropic MASS: GRAV, CENTRIF, ROTA.
emassd_anis_orient.inp	Anisotropic MASS with **ORIENTATION*: GRAV, CENTRIF, ROTA.
erotaryidr.inp	ROTARYI: ROTA.

Abaqus/Explicit element loading verification

Products: Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Gravity load

Elements tested

MASS T2D2 T3D2 B21 B31 PIPE21 PIPE31 SAX1 S3R S4R M3D3 M3D4R
CPE3 CPE4R CPS3 CPS4R CAX3 CAX4R C3D4 C3D6 C3D8R

Features tested

Gravity load and nonstructural mass.

Problem description

In this verification test all the available element types are tested by loading them with a gravity load. All the element nodes are fixed in position, and the reaction forces generated at the nodes are used to verify the element load calculations.

The material model is isotropic linear elasticity. The material properties used are defined as follows: Young's modulus = 193.1×10^9, Poisson's ratio = 0.3, and density = 7850.

A nonstructural mass contribution to the element mass is defined while the effective density is maintained at the above specified value by reducing the material density to the extent of the added nonstructural mass. Because the GRAV load is applied on both the structural mass and the nonstructural mass, the analytical solution used to verify the numerical results remains the same.

In the first step a gravity load is applied in the vertical direction (y-direction). The amplitude function for this gravity load is defined such that the load is ramped up to a value of 10 over the first half of the step and held constant over the second half of the step. In the second step the gravity load in the vertical direction is replaced with a gravity load in the horizontal direction (x-direction), which has an amplitude function that is similar to the vertical load.

Results and discussion

The results for all the elements agree with the analytical values, which are included at the top of the input file.

Input files

element_grav.inp	Input data used for this test.

Uniform body forces

Elements tested

T2D2 T3D2 SAX1 S3R S4R M3D3 M3D4R
CPE3 CPE4R CPS3 CPS4R CAX3 CAX4R C3D4 C3D6 C3D8R

Features tested

Uniform body forces.

Problem description

In this verification test all the available element types are tested by loading them with a uniform body force. All the element nodes are fixed in position, and the reaction forces generated at the nodes are used to verify the element load calculations.

The material model is isotropic linear elasticity. The material properties used are defined as follows: Young's modulus = 193.1×10^9, Poisson's ratio = 0.3, and density = 785.

In the first step a uniform body force of 1.0×10^5 is applied in the x-direction for all the elements except the axisymmetric elements, where it is applied in the r-direction. The amplitude function for this body force is defined such that the load is ramped on over the first half of the step and held constant for the rest of the analysis. In the second step another uniform body force of 1.0×10^5 is applied in the y-direction for all the elements except the axisymmetric elements, where it is applied in the z-direction. This load is applied using the same amplitude function that was used in the first step. For C3D4, C3D6, C3D8R, S3R, S4R, M3D3, and M3D4R elements, another uniform body force of 1.0×10^5 is applied in the z-direction in a third step. This load also has the same amplitude function that was used in the first step.

Results and discussion

The results for all the elements agree with the analytical values, which are included at the top of the input file.

Input files

element_body.inp	Input data used for this test.

Uniform pressure load

Elements tested

Distributed element-based loads

RAX2 R2D2 R3D3 R3D4 B21 B31 PIPE21 PIPE31 SAX1 S3R S4R M3D3 M3D4R CPE3 CPE4R CPS3 CPS4R CAX3 CAX4R C3D4 C3D6 C3D8R C3D10 C3D10M

Distributed surface loads

RAX2 R2D2 R3D3 R3D4 SAX1 S3R S4R M3D3 M3D4R CPE3 CPE4R CPS3 CPS4R CAX3 CAX4R C3D4 C3D6 C3D8R

Features tested

Uniform pressure load prescribed with distributed element-based and surface loads.

Problem description

In these verification tests all the available element types are tested by loading them with uniform pressure using distributed element-based loads and distributed surface loads. All the element nodes are fixed in position, and the reaction forces generated at the nodes are used to verify the load applications. Pipe elements (PIPE21 and PIPE31) are tested only with distributed element-based loads. Multiple steps are used to apply different loads. All the loads applied in previous steps are removed at the beginning of each step. Loads are linearly increased over the first half of each step and held constant over the second half.

Isotropic linearly elastic material is used for all elements. The material properties used are defined as follows: Young's modulus = 193.1×10^9, Poisson's ratio = 0.3, and density = 785.

For beam (B21, B31) and pipe (PIPE21, PIPE31) elements in the case of element-based loads, uniform distributed force per unit length of 1.0×10^5 is applied in along the x- and y-direction in the first and second steps, respectively. In the third step uniform distributed force per unit length of 1.0×10^5 along the z-direction is applied on three-dimensional beam (B31) and pipe (PIPE31) elements.

For shell elements (S3R, S4R) and axisymmetric line elements (SAX1) uniform distributed normal force per unit area of 1.0×10^5 is applied in the first step.

For three-edged planar elements (CPE3, CPE6M, CPS3, CPS6M) and axisymmetric elements (CAX3, CAX4R) a uniform distributed normal force per unit length of 1.0×10^5 is applied on each element edge in the first three steps.

For four-edged planar elements (CPE4R, CPS4R) and axisymmetric elements (CAX4R) a uniform distributed normal force per unit length of 1.0×10^5 is applied on each element edge in the first four steps.

For tetrahedral three-dimensional continuum elements (C3D4, C3D10, and C3D10M) a uniform distributed force per unit area of 1.0×10^5 is applied on each face in the first four steps.

For prismatic three-dimensional continuum elements (C3D6) a uniform distributed force per unit area of 1.0×10^5 is applied on each face in the first five steps.

For hexahedral three-dimensional continuum elements (C3D8) a uniform distributed force per unit area of 1.0×10^5 is applied on each face in the first six steps.

In the case of surface-based loads, in the first step a uniform pressure of 1.0×10^5 is applied on one of the element edge surfaces (for CPE3, CPE4R, CPS3, CPS4R, CAX3, CAX4R, SAX1, R2D2, and RAX2 elements) or element faces (for C3D4, C3D6, C3D8R, S3R, S4R, M3D3, M3D4R, R3D3, and R3D4 elements). In the second step the same uniform pressure is applied on other element edge surfaces or element faces.

Results and discussion

The results for all the elements agree with the analytical values, which are included at the top of the input file.

Input files

element_pres.inp	Input data for element-based pressure loads used for this test.
surface_pres.inp	Input data for surface-based pressure loads used for this test.

Viscous pressure load

Elements tested

Distributed element-based and surface loads

SAX1 S3R S4R M3D3 M3D4R CPE3 CPE4R CPS3 CPS4R CAX3 CAX4R C3D4 C3D6 C3D8R

Features tested

Viscous pressure load.

Problem description

In this verification test all the available element types are tested by loading them with a viscous pressure load. The nodes belonging to the plane strain, plane stress, and axisymmetric elements (CPE3, CPE4R, CPS3, CPS4R, CAX3, and CAX4R) are constrained in the x-direction; and an initial velocity of 100 is prescribed in the

y-direction. The nodes belonging to the three-dimensional elements (C3D4, C3D6, and C3D8R) are constrained in the *x*- and *z*-directions, and an initial velocity of 100 is prescribed in the *y*-direction. The nodes belonging to the shell and membrane elements (S3R, S4R, M3D3, and M3D4R) are constrained in the *x*- and *y*-directions, and an initial velocity of 100 is prescribed in the *z*-direction. The nodes belonging to the axisymmetric shell element (SAX1) are constrained in the *z*-direction, and an initial velocity of 100 is prescribed in the *r*-direction.

The material model is isotropic linear elasticity. The material properties used are defined as follows: Young's modulus = 193.1×10^9, Poisson's ratio = 0.3, and density = 7850. The coefficient of viscosity is 1000.

The viscous pressure load generates reaction forces at the nodes, which are used to verify the element load calculations. This test has only one step.

Results and discussion

The results for all the elements agree with the analytical values, which are included at the top of the input file.

Input files

element_vpres.inp	Input data for element-based loads used for this test.
surface_vpres.inp	Input data for surface-based loads used for this test.

Viscous body and stagnation loads

Elements tested

Distributed element-based and surface loads

SAX1 S3R S4R M3D3 M3D4R CPE3 CPE4R CPS3 CPS4R CAX3 CAX4R C3D4 C3D6 C3D8R

Features tested

Viscous body and stagnation loads.

Problem description

In this verification test all the available element types are tested by loading them with a viscous body or a stagnation load. The nodes belonging to the plane strain, plane stress, and axisymmetric elements (CPE3, CPE4R, CPS3, CPS4R, CAX3, and CAX4R) are constrained in the *x*-direction; and an initial velocity of 100 is prescribed in the *y*-direction. The nodes belonging to the three-dimensional elements (C3D4, C3D6, and C3D8R) are constrained in the *x*- and *z*-directions, and an initial velocity of 100 is prescribed in the *y*-direction. The nodes belonging to the shell and membrane elements (S3R, S4R, M3D3, and M3D4R) are constrained in the *x*- and *y*-directions, and an initial velocity of 100 is prescribed in the *z*-direction. The nodes belonging to the axisymmetric shell element (SAX1) are constrained in the *z*-direction, and an initial velocity of 100 is prescribed in the *r*-direction.

The material model is isotropic linear elasticity. The material properties used are defined as follows: Young's modulus = 193.1×10^9, Poisson's ratio = 0.3, and density = 7850.

The viscous body and stagnation loads generate reaction forces at the nodes, which are used to verify the element load calculations.

Results and discussion

Viscous body force loading provides an alternative way to define the mass-proportional damping as a function of relative velocities and a step-dependent damping coefficient. In the testing of viscous body force loading, the results agree with those obtained by using the mass-proportional damping with damping factor of 7.85.

Input files

element_vbf.inp	Input data for viscous body loads.
surface_sp.inp	Input data for surface-based stagnation pressure loads.
element_sp.inp	Input data for element-based stagnation pressure loads.
element_sbf.inp	Input data for stagnation body loads.

Surface tension loads

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Transformation of a cube into a sphere

Elements tested

C3D8H C3D8R EC3D8R

Features tested

Surface tension loads.

Problem description

This verification problem evaluates the ability of C3D8R, EC3D8R and C3D8H elements, under surface tension loads, to transform a unit cube into a sphere.

A unit cube is modeled by using 1000 C3D8R elements in Abaqus/Explicit and 1000 C3D8H elements in Abaqus/Standard, respectively. A surface tension load of 3.8×10^{-6} kN/mm is applied on each exterior face. In Abaqus/Explicit the cube contains a fluid modeled using a linear $U_s - U_p$ equation of state material. The material parameters used are $c_0 = 1805$ mm/s, s=0.0, and $\Gamma_0 = 0$. A linear (Newtonian) viscous shear viscosity of $\eta = 1.5 \times 10^{-8}$ Ns/mm^2 is used, and the material density is set to $\rho = 9.2 \times 10^{-5}$ g/mm^3. In Abaqus/Standard the cube is modeled using an elastic plastic material with a very small yield stress.

Figure 1 shows the initial configuration of the cube. *Figure 2* and *Figure 3* show the final equilibrium configuration of the cube as it is transformed to a sphere under the action of surface tension.

For the Eulerian analysis, the Eulerian domain is modeled using a $20 \times 20 \times 20$ mesh of EC3D8R elements. At the beginning of the analysis, a unit cube occupying a $10 \times 10 \times 10$ element region within the Eulerian domain is assigned a fluid volume fraction of one, indicating that the region is completely filled with fluid material. All other Eulerian elements are initially empty. A surface tension load of 3.8×10^{-6} kN/mm is applied on the exterior faces of the given Eulerian material instance automatically. The same material described previously for Abaqus/Explicit is used.

Figure 4 shows the initial configuration of the Eulerian domain and a cube that is filled with fluid material. *Figure 5* uses an isosurface view cut based on the output of volume fraction to show the final equilibrium configuration of the cube as it is transformed to a sphere under the action of surface tension.

Results and discussion

This test problem demonstrates surface tension loads to model the effects of surface tension. The analytical solution yields a sphere radius of 0.620, and the numerical solution produces a sphere radius of 0.6197 in Abaqus/Explicit and 0.6223 in Abaqus/Standard, respectively. Despite using a very coarse mesh, the results show excellent agreement with the analytical solutions.

Input files

Abaqus/Explicit input files

surft_c3d8r.inp	Surface tension loads of three-dimensional elements.
surft_ec3d8r.inp	Surface tension loads of Eulerian elements.

Abaqus/Standard input files

surft_c3d8h_std.inp	Surface tension loads of three-dimensional elements.

Figures

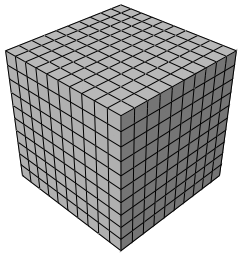


Figure 1: Initial configuration of the cube.

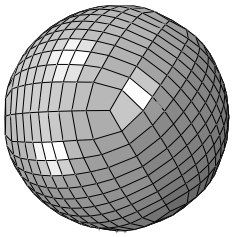


Figure 2: Final configuration of the cube in Abaqus/Explicit.

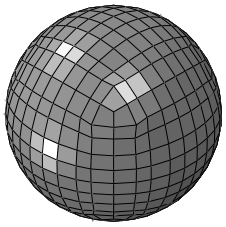


Figure 3: Final configuration of the cube in Abaqus/Standard.

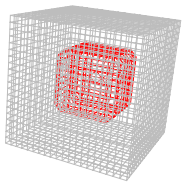


Figure 4: Eulerian domain and initial configuration of the cube.

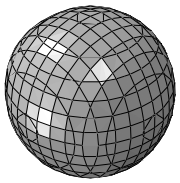


Figure 5: Final configuration of the cube with Eulerian elements.

Transformation of a square into a circle

Elements tested

CPE4H CPE4R CAX4H CAX4R

Features tested

Surface tension loads.

Problem description

This verification problem evaluates the ability of CPE4R, CAX4R, CPE4H, and CAX4H elements, under surface tension loads, to transform a unit square into a circle.

A unit square is modeled by using 1000 elements. A surface tension load of 3.8×10^{-6} kN/mm is applied on each exterior face. In Abaqus/Explicit the square contains a fluid modeled using a linear $U_s - U_p$ equation of state material. The material parameters used are $c_0 = 1805$ mm/s, s=0.0, and $\Gamma_0 = 0$. A linear (Newtonian) viscous shear viscosity is of $\eta = 1.5e^{-8}$ Ns/mm^2 is used, and the material density is set to $\rho = 9.2e^{-5}$ g/mm^3. In Abaqus/Standard the square is modeled using an elastic plastic material with a very small yield stress.

For the axisymmetric element CAX4R and CAX4H, only half of the square is modeled. The initial configuration of the square is shown in *Figure 6*. For plane strain elements the final equilibrium configuration of the square is shown in *Figure 7* and *Figure 8*. For axisymmetric elements the final equilibrium configuration of the square is shown in *Figure 9* and *Figure 10*.

Results and discussion

This test problem validates the effects of surface tension by applying surface tension loads. The analytical solutions yield a circle radius 0.564 for the two-dimensional case and 0.9085 for the axisymmetric case. The numerical solution from Abaqus/Explicit results in a circle radius of 0.5651 for the two-dimensional case and

0.9109 for the axisymmetric case. In Abaqus/Standard the numerical solution produces a circle radius of 0.565 for the two-dimensional case and 0.9111 for the axisymmetric case. Despite using very coarse meshes, both cases show excellent agreement with the analytical solutions.

Input files

Abaqus/Explicit input files

surft_cpe4r.inp	Surface tension loads of two-dimensional elements.
surft_cax4r.inp	Surface tension loads of axisymmetric elements.

Abaqus/Standard input files

surft_cpe4h_std.inp	Surface tension loads of two-dimensional elements.
surft_cax4h_std.inp	Surface tension loads of axisymmetric elements.

Figures

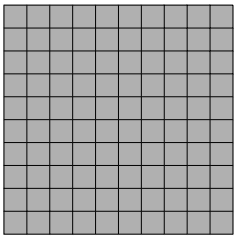


Figure 6: Initial configuration of the square.

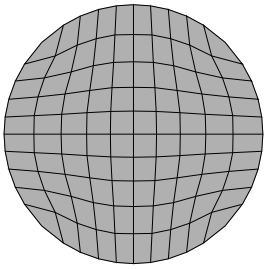


Figure 7: Final configuration of the square for two-dimensional elements in Abaqus/Explicit.

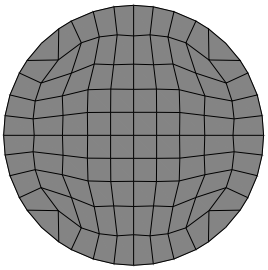


Figure 8: Final configuration of the square for two-dimensional elements in Abaqus/Standard.

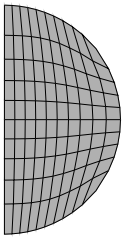


Figure 9: Final configuration of the square for axisymmetric elements in Abaqus/Explicit.

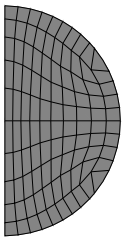


Figure 10: Final configuration of the square for axisymmetric elements in Abaqus/Standard.

Liquid droplet with fixed contact angles

Elements tested

C3D8H C3D8R CPE4H CPE4R CAX4H CAX4R

Features tested

Surface tension loads.

Problem description

This verification problem assesses fixed contact angles in a liquid droplet using different elements under surface tension loads.

The same models described previously are used to model a liquid droplet in contact with a wall and with fixed contact angles incorporated. A surface tension loads of $3.8e^{-6}\ kN/mm$ is applied on each exterior face except the bottom faces in contact with the wall. On the bottom face, a surface tension load of $-3.8e^{-6}\cos(\theta)$ is applied based on the specified contact angle and Young's equation. In addition, no-separation free-slip wall conditions are specified by applying a fixed boundary condition on degree of freedom 2 for nodes located on the bottom surface.

Results and discussion

Two fixed contact angles (60° and 120°) are applied separately in the simulation. Based on the final equilibrium configurations, the numerically computed contact angles are compared with their theoretical counterparts. Despite the use of a very coarse mesh, the numerical results closely match the theoretical values, as shown in *Table 1* and *Table 2*.

Table 1: Comparisons of theoretical and numerical contact angles in Abaqus/Explicit.

Elements	Theoretical Contact Angle	Numerical Contact Angles	Relative Errors
C3D8R	60°	58.36°	-2.73%
C3D8R	120°	117.97°	-1.69%
CPE4R	60°	58.62°	-2.3%
CPE4R	120°	115.18°	-4.02%
CAX4R	60°	59.30°	-1.17%
CAX4R	120°	116.15°	-3.20%

Table 2: Comparisons of theoretical and numerical values in Abaqus/Standard.

Elements	Theoretical Contact Angle	Numerical Contact Angles	Relative Errors
C3D8H	60°	57.56°	–4.07%
C3D8H	120°	116.83°	–2.64%
CPE4H	60°	57.46°	–4.23%
CPE4H	120°	116.06°	–3.29%
CAX4H	60°	58.51°	–2.48%
CAX4H	120°	115.13°	–4.06%

Input files

Abaqus/Explicit input files

surft_c3d8r_a60.inp — Surface tension loads of three-dimensional elements with contact angle 60°.

surft_c3d8r_a120.inp — Surface tension loads of three-dimensional elements with contact angle 120°.

surft_cpe4r_a60.inp — Surface tension loads of two-dimensional elements with contact angle 60°.

surft_cpe4r_a120.inp — Surface tension loads of two-dimensional elements with contact angle 120°.

surft_cax4r_a60.inp — Surface tension loads of axisymmetric elements with contact angle 60°.

surft_cax4r_a120.inp — Surface tension loads of axisymmetric elements with contact angle 120°.

Abaqus/Standard input files

surft_c3d8h_a60_std.inp — Surface tension loads of three-dimensional elements with contact angle 60°.

surft_c3d8h_a120_std.inp — Surface tension loads of three-dimensional elements with contact angle 120°.

surft_cpe4h_a60_std.inp — Surface tension loads of two-dimensional elements with contact angle 60°.

surft_cpe4h_a120_std.inp	Surface tension loads of two-dimensional elements with contact angle 120°.
surft_cax4h_a60_std.inp	Surface tension loads of axisymmetric elements with contact angle 60°.
surft_cax4h_a120_std.inp	Surface tension loads of axisymmetric elements with contact angle 120°.

A cubic solder droplet in contact with a pad

Elements tested

C3D8H C3D8R EC3D8R

Features tested

Surface tension loads.

Problem description

This verification problem examines the contact of a cubic solder droplet with a pad under surface tension loads.

A unit cubic solder is modeled using 1000 C3D8R elements in Abaqus/Explicit and 1000 C3D8H elements in Abaqus/Standard, respectively. A surface tension load of $4.985e^{-6}$ kN/mm is applied on each exterior face. For contact between the solder and the pad, a no-separation, free-slip condition is enforced. The contact angle is specified through a surface interaction definition that is assigned to the contact property.

In Abaqus/Explicit the melted state of the solder material is modeled using a linear $U_s - U_p$ equation of state material. The material parameters used are $c_0 = 1805$ mm/s, s=0.0, and $\Gamma_0 = 0$. A linear (Newtonian) viscous of $\eta = 1.5e^{-8}$ Ns/mm^2 is used, and the material density is set to $\rho = 8.93e^{-6}$ g/mm^3. In Abaqus/Standard the solder material is modeled using an elastic plastic material with a very small yield stress.

Figure 11 shows the initial configuration of the cubic solder and the pad. *Figure 12* and *Figure 13* show the final equilibrium configuration of the cubic solder and the pad. No initial velocity and gravity loads are considered in this model.

Results and discussion

Different contact angles are applied separately in the simulation. Based on the final equilibrium configurations, the numerically simulated values of height and width are computed and compared with their theoretical counterparts. Despite the use of a very coarse mesh, the numerical results closely match the theoretical values, as shown in *Table 3* and *Table 4*.

Table 3: Comparisons of theoretical and numerical values in Abaqus/Explicit.

Contact Angle	Theoretical Height	Numerical Height	Theoretical Width	Numerical Width
15°	0.222	0.2348 (5.67%)	3.376	3.499 (3.65%)
30°	0.355	0.3434 (-3.16%)	2.648	2.653 (0.21%)
45°	0.469	0.460 (-1.88%)	2.266	2.274 (0.35%)
105°	0.884	0.885 (0.11%)	1.404	1.408 (0.25%)
135°	1.080	1.082 (0.15%)	1.266	1.271 (0.45%)
150°	1.163	1.156 (–0.56%)	1.246	1.252 (0.50%)

Table 4: Comparisons of theoretical and numerical values in Abaqus/Standard.

Contact Angle	Theoretical Height	Numerical Height	Theoretical Width	Numerical Width
15°	0.222	0.2272 (2.34%)	3.376	3.3641 (–0.35%)
30°	0.355	0.3581 (0.87%)	2.648	2.6475 (–0.02%)
45°	0.469	0.4722 (0.68%)	2.266	2.2673 (0.06%)
105°	0.884	0.8876 (0.41%)	1.404	1.4064 (0.17%)
135°	1.080	1.0819 (0.18%)	1.266	1.27 (0.32%)
150°	1.163	1.1629 (–0.01%)	1.246	1.2517 (0.46%)

Input files

Abaqus/Explicit input files

solder_pad_a15.inp Surface tension loads with contact angle 15°.

solder_pad_a30.inp Surface tension loads with contact angle 30°.

solder_pad_a45.inp Surface tension loads with contact angle 45°.

solder_pad_a105.inp Surface tension loads with contact angle 105°.

solder_pad_a135.inp Surface tension loads with contact angle 135°.

solder_pad_a150.inp Surface tension loads with contact angle 150°.

solder_pad_ec3d8r_a15.inp EC3D8R elements with contact angle 15°.

solder_pad_ec3d8r_a30.inp EC3D8R elements with contact angle 30°.

solder_pad_ec3d8r_a135.inp EC3D8R elements with contact angle 135°.

Abaqus/Standard input files

solder_pad_a15_c3d8h_std.inp Surface tension loads with contact angle 15°.

solder_pad_a30_c3d8h_std.inp Surface tension loads with contact angle 30°.

solder_pad_a45_c3d8h_std.inp Surface tension loads with contact angle 45°.

solder_pad_a105_c3d8h_std.inp Surface tension loads with contact angle 105°.

solder_pad_a135_c3d8h_std.inp Surface tension loads with contact angle 135°.

solder_pad_a150_c3d8h_std.inp Surface tension loads with contact angle 150°.

Figures

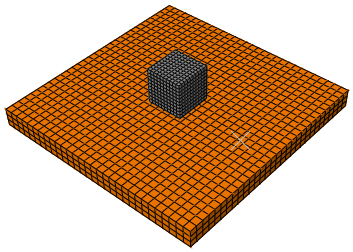


Figure 11: Initial configuration of the cubic solder and the pad.

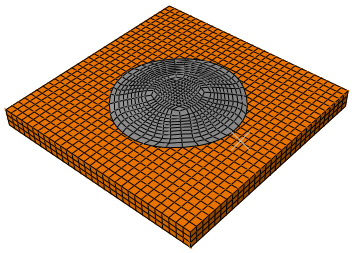


Figure 12: Final configuration of the solder and the pad for contact angle 15° case in Abaqus/Explicit.

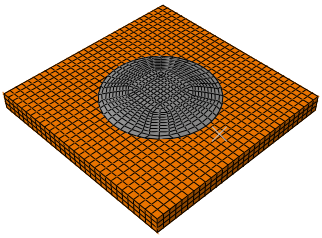


Figure 13: Final configuration of the solder and the pad for contact angle 15° case in Abaqus/Standard.

Two cubes coalescing into a single sphere

Elements tested

EC3D8R

Features tested

Surface tension loads.

Problem description

This verification problem evaluates the ability of EC3D8R elements to model the gradual coalescence of two droplets into a single spherical body under the action of surface tension loads.

The Eulerian domain is modeled using a 20 × 20 × 20 mesh of EC3D8R elements. Two identical cubes are initially separated by a small gap. Each cube initially occupies 10 × 10 × 10 elements within the Eulerian domain and is assigned a fluid volume fraction of one, indicating that the region is completely filled with fluid material. All other Eulerian elements are initially empty. A surface tension load of $\mathbf{3.8\times10^{-6}}$ kN/mm is applied on the exterior faces of the given Eulerian material instance. The cubes contain a fluid modeled using a linear $U_s - U_p$ equation of state material. The material parameters used are $c_0 = 1805$ mm/s, s=0.0, and $\Gamma_0 = 0$. A linear (Newtonian) viscous of $\eta = 1.5\times10^{-8}$ Ns/mm^2 is used, and the material density is set to $\rho = 9.2\times10^{-5}$ g/mm^3.

Results and discussion

Figure 14 uses an isosurface view cut based on the initial of volume fraction to show the initial configuration of two identical cubes. *Figure 15* uses an isosurface view cut based on the output of volume fraction to show the final equilibrium configuration of the cubes as they are transformed to a single sphere under the action of surface tension.

This test problem demonstrates how two initially separated material cubes modeled with Eulerian elements can merge and gradually evolve into a single spherical shape due to material flow and surface-energy minimization. It also demonstrates that the Eulerian formulation successfully captures severe deformation, material transport, interface evolution, and coalescence behavior without the mesh distortion problems typically encountered in purely Lagrangian methods.

This example is representative of typical large-deformation Eulerian simulations, such as droplet coalescence, molten metal merging, and additive manufacturing melt pools.

Input files

Abaqus/Explicit input files

surft_ec3d8r_twoballs.inp	Surface tension loads of Eulerian elements.

Figures

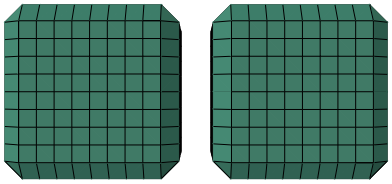


Figure 14: Initial configuration of two cubes with Eulerian elements.

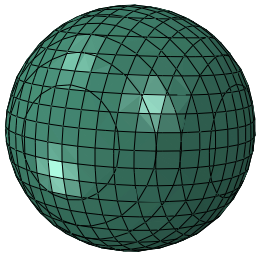


Figure 15: Final configuration of two cubes with Eulerian elements.

Two liquid droplets with fixed contact angles coalescing on a rigid pad

Elements tested

EC3D8R

Features tested

Surface tension loads.

Problem description

This verification problem assesses fixed contact angles in two liquid droplets using EC3D8R elements, under surface tension loads, gradually coalescing into a single body.

The problem demonstrates the coalescence of two droplets with a fixed contact angle on a rigid pad using Eulerian elements. Each droplet maintains a prescribed contact angle with solid surface during the coalescing process. The droplets are initially separated with a small gap and gradually merge into a single equilibrium droplet.

The liquid droplets are modeled in contact with a wall and with fixed contact angles incorporated. As in the previous example, the cubes contain a fluid modeled using a linear $U_s - U_p$ equation of state material. The material parameters used are $c_0 = 1805$ mm/s, s=0.0, and $\Gamma_0 = 0$. A linear (Newtonian) viscous of $\eta = 1.5\times10^{-8}$ Ns/mm^2 is used, and the material density is set to $\rho = 9.2\times10^{-6}$ g/mm^3. A surface tension load of 3.8×10^{-6} kN/mm is applied on the exterior faces of the given Eulerian material instance except for the material interfaces in contact with the wall. On the material interfaces in contact with the wall, a contact angle of 60° is specified by using a surface interaction definition that is assigned to the contact property. In addition, no-separation free-slip wall conditions are specified.

Results and discussion

Figure 16 uses an isosurface view cut based on the initial volume fraction on the pad to show the initial configuration of two identical cubes. *Figure 17* and *Figure 18* use an isosurface view cut based on the output of the volume fraction to show the early time and the final equilibrium configuration of the cubes, respectively, as they are transformed to a single sphere under the action of surface tension.

The simulation successfully captures droplet coalescence, free-surface evolution, and equilibrium wetting behavior.

Contact angles with Eulerian elements are enforced approximately due to the diffuse representation of material interfaces and the presence of numerical diffusion. Increasing the mesh resolution improves accuracy, but exact satisfaction of the prescribed contact angle is generally not expected.

Input files

Abaqus/Explicit input files

surft_ec3d8r_twoballs_a60.inp Two cubes on a rigid pad with a fixed contact angle.

Figures

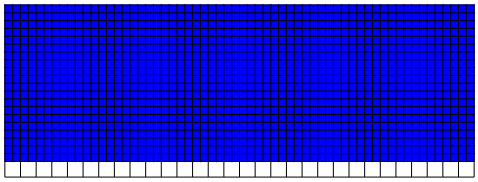


Figure 16: Initial configuration of two cubes on a rigid pad with Eulerian elements.

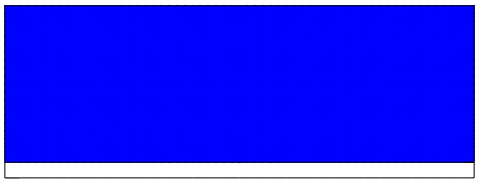


Figure 17: The configuration at an early time of two cubes on a rigid pad with Eulerian elements.

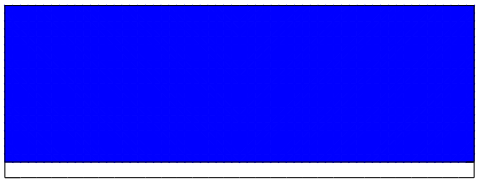


Figure 18: Final configuration of two cubes on a rigid pad with Eulerian elements.

Incident wave loading

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Features tested

- Incident wave
- Incident wave fluid properties
- Incident wave interaction
- Incident wave interaction property
- Incident wave loading formulation
- Incident wave properties
- Incident wave reflection

Acoustic element tests

Elements tested

AC2D3 AC2D4 AC2D4R AC2D6 AC2D8
AC3D4 AC3D5 AC3D6 AC3D8 AC3D8R AC3D10 AC3D15 AC3D20
ACAX3 ACAX4 ACAX4R ACAX6 ACAX8

Features tested

Incident wave loading on acoustic elements in Abaqus/Standard and Abaqus/Explicit.

Problem description

One-dimensional incident wave loading is tested in this verification set. The model consists of a column of fluid 1 m long with a square cross-section of area equal to 10^{-4}m^2. The length direction is the x-axis, while the cross-section is parallel to the y- and z-axes. In the axisymmetric case the column is oriented along the axial direction. The first-order element models consist of 100 elements for the quadrilateral cases and 200 elements for the triangular cases. The second-order element models consist of 50 and 100 elements for the quadrilateral and triangular cases, respectively. For all cases one element is used along the breadth and width directions.

A nonreflective boundary condition is imposed on one end of the column via impedance boundary conditions. The sound source is located at (−10, 0, 0) for the planar waves and at (−100000, 0, 0) for the spherical waves, while the standoff point is located at (0, 0, 0). The material properties of the fluid are the same as those of the surrounding medium. The material used is air with the following properties: density, $\rho_f = 1.21\ \text{kg/m}^3$; bulk modulus, $K_f = 1.424 \times 10^5$ Pa.

The sound source excitation is applied in two ways: through the pressure amplitude and through the corresponding acceleration amplitude. The pressure is applied as a ramp function beginning at zero and reaching a magnitude of 1.826 Pa at the end of 4.4 ms. The acceleration amplitude is applied through a step function with a magnitude of 1 m/s^2. Transient simulations are performed in both Abaqus/Standard and Abaqus/Explicit. The validity of

the solution is checked by comparing the POR value at the first node with the expected value of 1.826 Pa at the end of the step.

The total wave formulation option is also tested. The acoustic solution under the specified incident wave loading obtained using the total wave formulation option is compared to the acoustic solution obtained while using the default scattered wave formulation option.

A similar model is also created to test the bubble loading, with water used as the material instead of air.

Results and discussion

With the meshes used in these tests the result for all elements except AC3D4 is POR=1.825 Pa at node 1. The AC3D4 mesh yields a value of POR=1.865 Pa at node 1. Finer meshes yield more accurate results.

The results obtained using the total wave formulation option are found to be identical to those obtained using the default scattered wave formulation.

Input files

Abaqus/Standard input files

Planar wavefront, pressure amplitude:

iw_1d_ac2d3_dyl_p_pa.inp	AC2D3 elements.
iw_1d_ac2d4_dyl_p_pa.inp	AC2D4 elements.
iw_1d_ac2d6_dyl_p_pa.inp	AC2D6 elements.
iw_1d_ac2d8_dyl_p_pa.inp	AC2D8 elements.
iw_1d_ac3d4_dyl_p_pa.inp	AC3D4 elements.
iw_1d_ac3d5_dyl_p_pa.inp	AC3D5 elements.
iw_1d_ac3d6_dyl_p_pa.inp	AC3D6 elements.
iw_1d_ac3d8_dyl_p_pa.inp	AC3D8 elements.
iw_1d_ac3d10_dyl_p_pa.inp	AC3D10 elements.
iw_1d_ac3d15_dyl_p_pa.inp	AC3D15 elements.
iw_1d_ac3d20_dyl_p_pa.inp	AC3D20 elements.
iw_1d_acax3_dyl_p_pa.inp	ACAX3 elements.
iw_1d_acax4_dyl_p_pa.inp	ACAX4 elements.
iw_1d_acax6_dyl_p_pa.inp	ACAX6 elements.
iw_1d_acax8_dyl_p_pa.inp	ACAX8 elements.

Spherical wavefront, pressure amplitude:

iw_1d_ac2d3_dyl_s_pa.inp	AC2D3 elements.
iw_1d_ac2d4_dyl_s_pa.inp	AC2D4 elements.
iw_1d_ac2d6_dyl_s_pa.inp	AC2D6 elements.
iw_1d_ac2d8_dyl_s_pa.inp	AC2D8 elements.
iw_1d_ac3d4_dyl_s_pa.inp	AC3D4 elements.
iw_1d_ac3d5_dyl_s_pa.inp	AC3D5 elements.
iw_1d_ac3d6_dyl_s_pa.inp	AC3D6 elements.
iw_1d_ac3d8_dyl_s_pa.inp	AC3D8 elements.
iw_1d_ac3d10_dyl_s_pa.inp	AC3D10 elements.
iw_1d_ac3d15_dyl_s_pa.inp	AC3D15 elements.

iw_1d_ac3d20_dyl_s_pa.inp	AC3D20 elements.
iw_1d_acax3_dyl_s_pa.inp	ACAX3 elements.
iw_1d_acax4_dyl_s_pa.inp	ACAX4 elements.
iw_1d_acax6_dyl_s_pa.inp	ACAX6 elements.
iw_1d_acax8_dyl_s_pa.inp	ACAX8 elements.

Planar wavefront, acceleration amplitude:

iw_1d_ac2d3_dyl_p_aa.inp	AC2D3 elements.
iw_1d_ac2d4_dyl_p_aa.inp	AC2D4 elements.
iw_1d_ac2d6_dyl_p_aa.inp	AC2D6 elements.
iw_1d_ac2d8_dyl_p_aa.inp	AC2D8 elements.
iw_1d_ac3d4_dyl_p_aa.inp	AC3D4 elements.
iw_1d_ac3d5_dyl_p_aa.inp	AC3D5 elements.
iw_1d_ac3d6_dyl_p_aa.inp	AC3D6 elements.
iw_1d_ac3d8_dyl_p_aa.inp	AC3D8 elements.
iw_1d_ac3d10_dyl_p_aa.inp	AC3D10 elements.
iw_1d_ac3d15_dyl_p_aa.inp	AC3D15 elements.
iw_1d_ac3d20_dyl_p_aa.inp	AC3D20 elements.
iw_1d_acax3_dyl_p_aa.inp	ACAX3 elements.
iw_1d_acax4_dyl_p_aa.inp	ACAX4 elements.
iw_1d_acax6_dyl_p_aa.inp	ACAX6 elements.
iw_1d_acax8_dyl_p_aa.inp	ACAX8 elements.

Bubble-loading amplitude:

iw_1d_ac2d3_dyl_b_pa.inp	AC2D3 elements.
bubbledrag_iwi.inp	S4 elements, **INCIDENT WAVE INTERACTION* (preferred interface).
bubbledrag_iw.inp	S4 elements, **INCIDENT WAVE* (alternative interface).

Abaqus/Explicit input files

Planar wavefront, pressure amplitude:

iw_1d_ac2d3_xpl_p_pa.inp	AC2D3 elements.
iw_1d_ac2d4r_xpl_p_pa.inp	AC2D4R elements.
iw_1d_ac3d4_xpl_p_pa.inp	AC3D4 elements.
iw_1d_ac3d6_xpl_p_pa.inp	AC3D6 elements.
iw_1d_ac3d8r_xpl_p_pa.inp	AC3D8R elements.
iw_1d_acax3_xpl_p_pa.inp	ACAX3 elements.
iw_1d_acax4r_xpl_p_pa.inp	ACAX4R elements.
iwt_1d_ac2d4r_xpl_p_pa.inp	AC2D4R elements, total wave formulation.
iwt_1d_ac3d6_xpl_p_pa.inp	AC3D6 elements, total wave formulation.
iwt_1d_acax4r_xpl_p_pa.inp	ACAX4R elements, total wave formulation.

Spherical wavefront, pressure amplitude:

iw_1d_ac2d3_xpl_s_pa.inp	AC2D3 elements.
iw_1d_ac2d4r_xpl_s_pa.inp	AC2D4R elements.

iw_1d_ac3d4_xpl_s_pa.inp	AC3D4 elements.
iw_1d_ac3d6_xpl_s_pa.inp	AC3D6 elements.
iw_1d_ac3d8r_xpl_s_pa.inp	AC3D8R elements.
iw_1d_acax3_xpl_s_pa.inp	ACAX3 elements.
iw_1d_acax4r_xpl_s_pa.inp	ACAX4R elements.
iwt_1d_ac2d3_xpl_s_pa.inp	AC2D3 elements, total wave formulation.
iwt_1d_ac3d8r_xpl_s_pa.inp	AC3D8R elements, total wave formulation.
iwt_1d_acax3_xpl_s_pa.inp	ACAX3 elements, total wave formulation.

Planar wavefront, acceleration amplitude:

iw_1d_ac2d3_xpl_p_aa.inp	AC2D3 elements.
iw_1d_ac2d4r_xpl_p_aa.inp	AC2D4R elements.
iw_1d_ac3d4_xpl_p_aa.inp	AC3D4 elements.
iw_1d_ac3d6_xpl_p_aa.inp	AC3D6 elements.
iw_1d_ac3d8r_xpl_p_aa.inp	AC3D8R elements.
iw_1d_acax3_xpl_p_aa.inp	ACAX3 elements.
iw_1d_acax4r_xpl_p_aa.inp	ACAX4R elements.
iwt_1d_ac2d3_xpl_p_aa.inp	AC2D3 elements, total wave formulation.
iwt_1d_ac3d4_xpl_p_aa.inp	AC3D4 elements, total wave formulation.
iwt_1d_ac3d8r_xpl_p_aa.inp	AC3D8R elements, total wave formulation.
iwt_1d_acax3_xpl_p_aa.inp	ACAX3 elements, total wave formulation.

Bubble-loading amplitude:

iw_1d_ac2d3_xpl_b_pa.inp	AC2D3 elements.

Spherical, generalized decay:

iw_1d_ac3d4_xpl_s_pa_gendecay.inp	AC3D4 elements; generalized decay.
iw_1d_ac3d8r_xpl_s_pa_gendecay.inp	AC3D8R elements; generalized decay.
iw_aco_fastdecay.inp	AC3D8R elements; high spatial decay.
iw_aco_highc.inp	AC3D8R elements; high speed of sound.
iw_aco_nearlyacoustic.inp	AC3D8R elements; generalized, but nearly acoustic, decay.
iw_aco_slowdecay.inp	AC3D8R elements; generalized, but very slow, decay.
iw_b31_fastdecay.inp	B31 elements; high spatial decay.
iw_b31_highc.inp	B31 elements; high speed of sound.
iw_b31_nearlyacoustic.inp	B31 elements; generalized, but nearly acoustic, decay.
iw_b31_slowdecay.inp	B31 elements; generalized, but very slow, decay.
iw_cpl_fastdecay.inp	Coupled S4R and AC3D8R elements; high spatial decay.
iw_cpl_highc.inp	Coupled S4R and AC3D8R elements; high speed of sound.
iw_cpl_nearlyacoustic.inp	Coupled S4R and AC3D8R elements; generalized, but nearly acoustic, decay.
iw_cpl_nearlyacoustic0.inp	Coupled S4R and AC3D8R elements; generalized, but nearly acoustic, decay. Second test with different model properties.
iw_cpl_slowdecay.inp	Coupled S4R and AC3D8R elements; generalized, but very slow, decay.
iw_shl_fastdecay.inp	S4R elements; high spatial decay.

iw_shl_highc.inp	S4R elements; high speed of sound.
iw_shl_nearlyacoustic.inp	S4R elements; generalized, but nearly acoustic, decay.
iw_shl_slowdecay.inp	S4R elements; generalized, but very slow, decay.

Initialization of acoustic fields

Elements tested

AC2D3 AC2D4 AC2D4R AC2D6 AC2D8
AC3D4 AC3D5 AC3D6 AC3D8 AC3D8R AC3D10 AC3D15 AC3D20
ACAX3 ACAX4 ACAX4R ACAX6 ACAX8

Features tested

Incident wave loading on acoustic elements using incident wave loads and the total wave formulation in Abaqus/Standard and Abaqus/Explicit.

Problem description

These are multiple-element tests that model sound sources of planar waves and spherical waves exciting traveling waves in ducts. Two cases are studied: a spherical wave source using an exponentially decaying time amplitude and a plane wave source using a sinusoidal amplitude. In both cases the total wave formulation is used and the standoff point of the incident wave loading is specified to be inside the finite element mesh. Consequently, at the start of the analysis the incident waves have already travelled into the finite element domain. These tests show that at the start of the first dynamic step in the analysis the acoustic field is properly initialized to the values of the incident wave field.

Results and discussion

The results match the expected values for all cases.

Input files

Abaqus/Standard input files

std_twinit_2d_dcay.inp	Decay amplitude with spherical wavefront; two-dimensional elements.
std_twinit_3d_dcay.inp	Decay amplitude with spherical wavefront; three-dimensional elements.
std_twinit_ax_dcay.inp	Decay amplitude with spherical wavefront; axisymmetric elements.
std_twinit_2d_sine.inp	Sinusoidal amplitude with planar wavefront; two-dimensional elements.
std_twinit_3d_sine.inp	Sinusoidal amplitude with planar wavefront; three-dimensional elements.
std_twinit_ax_sine.inp	Sinusoidal amplitude with planar wavefront; axisymmetric elements.

Abaqus/Explicit input files

xpl_twinit_2d_dcay.inp	Decay amplitude with spherical wavefront; two-dimensional elements.
xpl_twinit_3d_dcay.inp	Decay amplitude with spherical wavefront; three-dimensional elements.
xpl_twinit_ax_dcay.inp	Decay amplitude with spherical wavefront; axisymmetric elements.
xpl_twinit_2d_sine.inp	Sinusoidal amplitude with planar wavefront; two-dimensional elements.
xpl_twinit_3d_sine.inp	Sinusoidal amplitude with planar wavefront; three-dimensional elements.
xpl_twinit_ax_sine.inp	Sinusoidal amplitude with planar wavefront; axisymmetric elements.

Beam element tests

Elements tested

B21 B21H B22 B22H B23 B23H B31 PIPE21 PIPE31

Features tested

Incident wave loading on two-dimensional beam elements in Abaqus/Standard and Abaqus/Explicit. Pipe elements and three-dimensional beams are also tested in Abaqus/Explicit.

Problem description

In the case of two-dimensional modeling single-element tests are used to verify incident wave loading on two-dimensional beam and pipe elements, where the wave source is located at (0.5, 10) for the planar waves and at (0.5, 100000) for the spherical waves. The single element for each case is placed along the x-axis with end points at (0, 0) and (1, 0). All nodes are completely fixed. The standoff point is at (0.5, 0). The beam element has a square cross-section of area 1×10^{-4} m^2, whereas the pipe has an outer diameter of 1.0×10^{-2} m and the thickness of 1.0×10^{-3} m. The material properties for the beam are $\boldsymbol{E} = 1.0 \times 10^6$ Pa and $\boldsymbol{\rho} = 1000$ kg/m^3. The properties of the surrounding medium are the same as those used in the previous section.

The loading is applied as a ramp function with a maximum value of 1000 Pa attained at the end of the step at 0.5 ms. The reaction forces at the element nodes are compared. The expected reaction force at each of the end nodes is 500 N for the linear elements. For quadratic elements the expected reaction force is 166.7 N at each of the end nodes and 666.7 N at the mid node.

The bubble loading is also tested but with water used as the material instead of air.

In the case of three-dimensional modeling for verification on three-dimensional beam and pipe elements in Abaqus/Explicit, a beam comprised of 50 beam (B31) or pipe (PIPE31) elements, placed along the x-axis with end points (−50, 0,0) and (50,0,0) is used. In both cases a cross-section of type pipe with an outer diameter of 2.5 m and a thickness of 0.1 m is used. The material properties for the beam are $\boldsymbol{E} = 2.0 \times 10^{11}$ Pa and $\boldsymbol{\rho} = 10000$ kg/m^3.

The source of the spherical wave, due to an under water explosion, is located at (0,−30,0); and the stand-off point is located at (0,−5,0). The wave load is applied over a cylindrical skin modeled with surface membrane elements (SFM3D4R) that is tied to the outer surface of the beam. Boundary conditions on the beam disallow any axial displacement and rotations along the y- and z-axis. The solution is computed for 0.1s.

Results and discussion

The results exactly match the expected values for all cases.

Input files

Abaqus/Standard input files

Planar wavefront, geometric nonlinearities are ignored:

iw_1d_b21_dyl_p_pp.inp	B21 element.
iw_1d_b21h_dyl_p_pp.inp	B21H element.
iw_1d_b22_dyl_p_pp.inp	B22 element.
iw_1d_b22h_dyl_p_pp.inp	B22H element.
iw_1d_b23_dyl_p_pp.inp	B23 element.
iw_1d_b23h_dyl_p_pp.inp	B23H element.

Spherical wavefront, geometric nonlinearities are ignored:

iw_1d_b21_dyl_s_pp.inp	B21 element.
iw_1d_b21h_dyl_s_pp.inp	B21H element.
iw_1d_b22_dyl_s_pp.inp	B22 element.
iw_1d_b22h_dyl_s_pp.inp	B22H element.
iw_1d_b23_dyl_s_pp.inp	B23 element.
iw_1d_b23h_dyl_s_pp.inp	B23H element.

Planar wavefront, geometric nonlinearities are considered:

iw_1d_b21_dyn_p_pp.inp	B21 element.
iw_1d_b21h_dyn_p_pp.inp	B21H element.
iw_1d_b22_dyn_p_pp.inp	B22 element.
iw_1d_b22h_dyn_p_pp.inp	B22H element.
iw_1d_b23_dyn_p_pp.inp	B23 element.
iw_1d_b23h_dyn_p_pp.inp	B23H element.

Spherical wavefront, geometric nonlinearities are considered:

iw_1d_b21_dyn_s_pp.inp	B21 element.
iw_1d_b21h_dyn_s_pp.inp	B21H element.
iw_1d_b22_dyn_s_pp.inp	B22 element.
iw_1d_b22h_dyn_s_pp.inp	B22H element.
iw_1d_b23_dyn_s_pp.inp	B23 element.
iw_1d_b23h_dyn_s_pp.inp	B23H element.

Bubble-loading amplitude:

iw_1d_b21_dyl_b_pp.inp	B21 element.

Abaqus/Explicit input files

iw_1d_b21_xpl_p_pp.inp	B21 element with planar wavefront.
iw_1d_b21_xpl_s_pp.inp	B21 element with spherical wavefront.
iwt_1d_b21_xpl_p_pp.inp	B21 element with planar wavefront.
iwt_1d_b21_xpl_s_pp.inp	B21 element with spherical wavefront.
iw_1d_p21_xpl_p_pp.inp	PIPE21 element with planar wavefront.
b31_sfm_iw.inp	B31 element with spherical wavefront.
p31_sfm_iw.inp	PIPE31 element with spherical wavefront.

Bubble-loading amplitude:

iw_1d_b21_xpl_b_pp.inp	B21 element.

Shell element tests

Elements tested

S3R S3RS S4R S4R5 S4RS S4RSW S8R S8R5 S9R5 STRI3 STRI65 SAX1 SAX2

Features tested

Incident wave loading on shell elements in Abaqus/Standard and Abaqus/Explicit.

Problem description

These are single-element tests that model a sound source at (0.5, 0.5, 10) for the planar shells and at (0, −10) for the axisymmetric shells for the planar waves. For the spherical waves the source is moved to (0.5, 0.5, 100000) for the planar shells and to (0, −100000) for the axisymmetric shells. The planar shell is modeled to be in the *X*–*Y* plane with unit length on all sides. The standoff point is located at (0.5, 0.5, 0). In the axisymmetric case the shell is oriented along the radial direction and the standoff point is at (0, 0). The shell thickness is 10^{-4}m. The shell material properties are the same as those of the beam in the previous section. The properties of the surrounding medium are kept the same as those used in the previous cases. All nodes are fixed completely. The loading is applied as a ramp function attaining a maximum of 1000 Pa at the end of the step at 0.5 ms. The reaction forces are compared with the expected values, which when summed should produce a total force of 1000 N.

A similar model is also created to test the bubble loading, with water used as the material instead of air.

Results and discussion

The results for all tested elements exactly match the expected values.

Input files

Abaqus/Standard input files

Planar wavefront, geometric nonlinearities are ignored:

iw_1d_s3r_dyl_p_pp.inp	S3R element.
iw_1d_s4_dyl_p_pp.inp	S4 element.
iw_1d_s4r_dyl_p_pp.inp	S4R element.
iw_1d_s4r5_dyl_p_pp.inp	S4R5 element.
iw_1d_s8r_dyl_p_pp.inp	S8R element.
iw_1d_s8r5_dyl_p_pp.inp	S8R5 element.
iw_1d_s9r5_dyl_p_pp.inp	S9R5 element.
iw_1d_stri3_dyl_p_pp.inp	STRI3 element.
iw_1d_stri65_dyl_p_pp.inp	STRI65 element.
iw_1d_sax1_dyl_p_pp.inp	SAX1 element.
iw_1d_sax2_dyl_p_pp.inp	SAX2 element.

Spherical wavefront, geometric nonlinearities are ignored:

iw_1d_s3r_dyl_s_pp.inp	S3R element.
iw_1d_s4_dyl_s_pp.inp	S4 element.
iw_1d_s4r_dyl_s_pp.inp	S4R element.
iw_1d_s4r5_dyl_s_pp.inp	S4R5 element.
iw_1d_s8r_dyl_s_pp.inp	S8R element.
iw_1d_s8r5_dyl_s_pp.inp	S8R5 element.
iw_1d_s9r5_dyl_s_pp.inp	S9R5 element.
iw_1d_stri3_dyl_s_pp.inp	STRI3 element.
iw_1d_stri65_dyl_s_pp.inp	STRI65 element.

iw_1d_sax1_dyl_s_pp.inp	SAX1 element.
iw_1d_sax2_dyl_s_pp.inp	SAX2 element.

Planar wavefront, geometric nonlinearities are considered:

iw_1d_s3r_dyn_p_pp.inp	S3R element.
iw_1d_s4_dyn_p_pp.inp	S4 element.
iw_1d_s4r_dyn_p_pp.inp	S4R element.
iw_1d_s4r5_dyn_p_pp.inp	S4R5 element.
iw_1d_s8r_dyn_p_pp.inp	S8R element.
iw_1d_s8r5_dyn_p_pp.inp	S8R5 element.
iw_1d_s9r5_dyn_p_pp.inp	S9R5 element.
iw_1d_stri3_dyn_p_pp.inp	STRI3 element.
iw_1d_stri65_dyn_p_pp.inp	STRI65 element.
iw_1d_sax1_dyn_p_pp.inp	SAX1 element.
iw_1d_sax2_dyn_p_pp.inp	SAX2 element.

Spherical wavefront, geometric nonlinearities are considered:

iw_1d_s3r_dyn_s_pp.inp	S3R element.
iw_1d_s4_dyn_s_pp.inp	S4 element.
iw_1d_s4r_dyn_s_pp.inp	S4R element.
iw_1d_s4r5_dyn_s_pp.inp	S4R5 element.
iw_1d_s8r_dyn_s_pp.inp	S8R element.
iw_1d_s8r5_dyn_s_pp.inp	S8R5 element.
iw_1d_s9r5_dyn_s_pp.inp	S9R5 element.
iw_1d_stri3_dyn_s_pp.inp	STRI3 element.
iw_1d_stri65_dyn_s_pp.inp	STRI65 element.
iw_1d_sax1_dyn_s_pp.inp	SAX1 element.
iw_1d_sax2_dyn_s_pp.inp	SAX2 element.

Bubble-loading amplitude:

iw_1d_s4_dyl_b_pp.inp	S4 element.

Abaqus/Explicit input files

Planar wavefront:

iw_1d_s3r_xpl_p_pp.inp	S3R element.
iw_1d_s3rs_xpl_p_pp.inp	S3RS element.
iw_1d_s4r_xpl_p_pp.inp	S4R element.
iw_1d_s4rs_xpl_p_pp.inp	S4RS element.
iw_1d_s4rsw_xpl_p_pp.inp	S4RSW element.
iw_1d_sax1_xpl_p_pp.inp	SAX1 element.
iwt_1d_sax1_xpl_p_pp.inp	SAX1 element.

Spherical wavefront:

iw_1d_s3r_xpl_s_pp.inp	S3R element.
iw_1d_s3rs_xpl_s_pp.inp	S3RS element.

iw_1d_s4r_xpl_s_pp.inp	S4R element.
iw_1d_s4rs_xpl_s_pp.inp	S4RS element.
iw_1d_s4rsw_xpl_s_pp.inp	S4RSW element.
iw_1d_sax1_xpl_s_pp.inp	SAX1 element.
iwt_1d_s3rs_xpl_s_pp.inp	S3RS element.
iwt_1d_s4r_xpl_s_pp.inp	S4R element.

Bubble-loading amplitude:

iw_1d_s4r_xpl_b_pp.inp	S4R element.

Solid element tests

Elements tested

CPE3 CPE4I CPE4R CPEG4I CPEG4R CPE6M CPEG6M CPE8 CPEG8
CPS3 CPS4I CPS4R CPS6 CPS6M CPS8R
C3D4 C3D5 C3D6 C3D8I C3D8R C3D10M C3D15V C3D20
CAX3 CAX4R CAX6 CAX6M CAX8R

Features tested

Incident wave loading on solid elements in Abaqus/Standard and Abaqus/Explicit.

Problem description

These tests use exactly the same geometry as that used in the acoustic element tests, except that the length is reduced to 0.1 m. Consequently, 10 and 20 first-order elements are used in the quadrilateral and triangular cases, respectively; and 5 and 10 second-order elements are used for the quadrilateral and triangular cases, respectively. The sound source is at (−10, 0) for the planar waves and at (−100000, 0, 0) for the spherical waves. All nodes are fixed in the y-direction, while the end nodes on the surface further away from the source are fixed additionally in the x-direction. The stresses in the elements are compared with those obtained using the equivalent distributed surface load.

A similar model is also created to test the bubble loading, with water used as the material instead of air.

Results and discussion

The solution is exactly the same as that obtained using the equivalent distributed surface load, except for the CPE6M element which gives a small percentage of error in the Abaqus/Explicit analysis.

Input files

Abaqus/Standard input files

Planar wavefront, geometric nonlinearities are ignored:

iw_1d_cpe3_dyl_p_pp.inp	CPE3 elements.
iw_1d_cpe4i_dyl_p_pp.inp	CPE4I elements.
iw_1d_cpe4r_dyl_p_pp.inp	CPE4R elements.
iw_1d_cpeg4i_dyl_p_pp.inp	CPEG4I elements.
iw_1d_cpeg4r_dyl_p_pp.inp	CPEG4R elements.
iw_1d_cpe6m_dyl_p_pp.inp	CPE6M elements.
iw_1d_cpeg6m_dyl_p_pp.inp	CPEG6M elements.

iw_1d_cpe8_dyl_p_pp.inp	CPE8 elements.
iw_1d_cpeg8_dyl_p_pp.inp	CPEG8 elements.
iw_1d_cps3_dyl_p_pp.inp	CPS3 elements.
iw_1d_cps4i_dyl_p_pp.inp	CPS4I elements.
iw_1d_cps4r_dyl_p_pp.inp	CPS4R elements.
iw_1d_cps6_dyl_p_pp.inp	CPS6 elements.
iw_1d_cps6m_dyl_p_pp.inp	CPS6M elements.
iw_1d_cps8r_dyl_p_pp.inp	CPS8R elements.
iw_1d_c3d4_dyl_p_pp.inp	C3D4 elements.
iw_1d_c3d5_dyl_p_pp.inp	C3D5 elements.
iw_1d_c3d6_dyl_p_pp.inp	C3D6 elements.
iw_1d_c3d8i_dyl_p_pp.inp	C3D8I elements.
iw_1d_c3d8r_dyl_p_pp.inp	C3D8R elements.
iw_1d_c3d10m_dyl_p_pp.inp	C3D10M elements.
iw_1d_c3d15v_dyl_p_pp.inp	C3D15V elements.
iw_1d_c3d20_dyl_p_pp.inp	C3D20 elements.
iw_1d_cax3_dyl_p_pp.inp	CAX3 elements.
iw_1d_cax4r_dyl_p_pp.inp	CAX4R elements.
iw_1d_cax6_dyl_p_pp.inp	CAX6 elements.
iw_1d_cax6m_dyl_p_pp.inp	CAX6M elements.
iw_1d_cax8r_dyl_p_pp.inp	CAX8R elements.

Spherical wavefront, geometric nonlinearities are ignored:

iw_1d_cpe3_dyl_s_pp.inp	CPE3 elements.
iw_1d_cpe4i_dyl_s_pp.inp	CPE4I elements.
iw_1d_cpe4r_dyl_s_pp.inp	CPE4R elements.
iw_1d_cpeg4i_dyl_s_pp.inp	CPEG4I elements.
iw_1d_cpeg4r_dyl_s_pp.inp	CPEG4R elements.
iw_1d_cpe6m_dyl_s_pp.inp	CPE6M elements.
iw_1d_cpeg6m_dyl_s_pp.inp	CPEG6M elements.
iw_1d_cpe8_dyl_s_pp.inp	CPE8 elements.
iw_1d_cpeg8_dyl_s_pp.inp	CPEG8 elements.
iw_1d_cps3_dyl_s_pp.inp	CPS3 elements.
iw_1d_cps4i_dyl_s_pp.inp	CPS4I elements.
iw_1d_cps4r_dyl_s_pp.inp	CPS4R elements.
iw_1d_cps6_dyl_s_pp.inp	CPS6 elements.
iw_1d_cps6m_dyl_s_pp.inp	CPS6M elements.
iw_1d_cps8r_dyl_s_pp.inp	CPS8R elements.
iw_1d_c3d4_dyl_s_pp.inp	C3D4 elements.
iw_1d_c3d5_dyl_s_pp.inp	C3D5 elements.
iw_1d_c3d6_dyl_s_pp.inp	C3D6 elements.
iw_1d_c3d8i_dyl_s_pp.inp	C3D8I elements.
iw_1d_c3d8r_dyl_s_pp.inp	C3D8R elements.

iw_1d_c3d10m_dyl_s_pp.inp	C3D10M elements.
iw_1d_c3d15v_dyl_s_pp.inp	C3D15V elements.
iw_1d_c3d20_dyl_s_pp.inp	C3D20 elements.
iw_1d_cax3_dyl_s_pp.inp	CAX3 elements.
iw_1d_cax4r_dyl_s_pp.inp	CAX4R elements.
iw_1d_cax6_dyl_s_pp.inp	CAX6 elements.
iw_1d_cax6m_dyl_s_pp.inp	CAX6M elements.
iw_1d_cax8r_dyl_s_pp.inp	CAX8R elements.

Planar wavefront, geometric nonlinearities are considered:

iw_1d_cpe3_dyn_p_pp.inp	CPE3 elements.
iw_1d_cpe4i_dyn_p_pp.inp	CPE4I elements.
iw_1d_cpe4r_dyn_p_pp.inp	CPE4R elements.
iw_1d_cpeg4i_dyn_p_pp.inp	CPEG4I elements.
iw_1d_cpeg4r_dyn_p_pp.inp	CPEG4R elements.
iw_1d_cpe6m_dyn_p_pp.inp	CPE6M elements.
iw_1d_cpeg6m_dyn_p_pp.inp	CPEG6M elements.
iw_1d_cpe8_dyn_p_pp.inp	CPE8 elements.
iw_1d_cpeg8_dyn_p_pp.inp	CPEG8 elements.
iw_1d_cps3_dyn_p_pp.inp	CPS3 elements.
iw_1d_cps4i_dyn_p_pp.inp	CPS4I elements.
iw_1d_cps4r_dyn_p_pp.inp	CPS4R elements.
iw_1d_cps6_dyn_p_pp.inp	CPS6 elements.
iw_1d_cps6m_dyn_p_pp.inp	CPS6M elements.
iw_1d_cps8r_dyn_p_pp.inp	CPS8R elements.
iw_1d_c3d4_dyn_p_pp.inp	C3D4 elements.
iw_1d_c3d5_dyn_p_pp.inp	C3D5 elements.
iw_1d_c3d6_dyn_p_pp.inp	C3D6 elements.
iw_1d_c3d8i_dyn_p_pp.inp	C3D8I elements.
iw_1d_c3d8r_dyn_p_pp.inp	C3D8R elements.
iw_1d_c3d10m_dyn_p_pp.inp	C3D10M elements.
iw_1d_c3d15v_dyn_p_pp.inp	C3D15V elements.
iw_1d_c3d20_dyn_p_pp.inp	C3D20 elements.
iw_1d_cax3_dyn_p_pp.inp	CAX3 elements.
iw_1d_cax4r_dyn_p_pp.inp	CAX4R elements.
iw_1d_cax6_dyn_p_pp.inp	CAX6 elements.
iw_1d_cax6m_dyn_p_pp.inp	CAX6M elements.
iw_1d_cax8r_dyn_p_pp.inp	CAX8R elements.

Spherical wavefront, geometric nonlinearities are considered:

iw_1d_cpe3_dyn_s_pp.inp	CPE3 elements.
iw_1d_cpe4i_dyn_s_pp.inp	CPE4I elements.
iw_1d_cpe4r_dyn_s_pp.inp	CPE4R elements.
iw_1d_cpeg4i_dyn_s_pp.inp	CPEG4I elements.

iw_1d_cpeg4r_dyn_s_pp.inp	CPEG4R elements.
iw_1d_cpe6m_dyn_s_pp.inp	CPE6M elements.
iw_1d_cpeg6m_dyn_s_pp.inp	CPEG6M elements.
iw_1d_cpe8_dyn_s_pp.inp	CPE8 elements.
iw_1d_cpeg8_dyn_s_pp.inp	CPEG8 elements.
iw_1d_cps3_dyn_s_pp.inp	CPS3 elements.
iw_1d_cps4i_dyn_s_pp.inp	CPS4I elements.
iw_1d_cps4r_dyn_s_pp.inp	CPS4R elements.
iw_1d_cps6_dyn_s_pp.inp	CPS6 elements.
iw_1d_cps6m_dyn_s_pp.inp	CPS6M elements.
iw_1d_cps8r_dyn_s_pp.inp	CPS8R elements.
iw_1d_c3d4_dyn_s_pp.inp	C3D4 elements.
iw_1d_c3d5_dyn_s_pp.inp	C3D5 elements.
iw_1d_c3d6_dyn_s_pp.inp	C3D6 elements.
iw_1d_c3d8i_dyn_s_pp.inp	C3D8I elements.
iw_1d_c3d8r_dyn_s_pp.inp	C3D8R elements.
iw_1d_c3d10m_dyn_s_pp.inp	C3D10M elements.
iw_1d_c3d15v_dyn_s_pp.inp	C3D15V elements.
iw_1d_c3d20_dyn_s_pp.inp	C3D20 elements.
iw_1d_cax3_dyn_s_pp.inp	CAX3 elements.
iw_1d_cax4r_dyn_s_pp.inp	CAX4R elements.
iw_1d_cax6_dyn_s_pp.inp	CAX6 elements.
iw_1d_cax6m_dyn_s_pp.inp	CAX6M elements.
iw_1d_cax8r_dyn_s_pp.inp	CAX8R elements.

Bubble-loading amplitude:

iw_1d_cpe4r_dyl_b_pp.inp	CPE4R elements.

Abaqus/Explicit input files

Planar wavefront:

iw_1d_cpe3_xpl_p_pp.inp	CPE3 elements.
iw_1d_cpe4r_xpl_p_pp.inp	CPE4R elements.
iw_1d_cpe6m_xpl_p_pp.inp	CPE6M elements.
iw_1d_cps3_xpl_p_pp.inp	CPS3 elements.
iw_1d_cps4r_xpl_p_pp.inp	CPS4R elements.
iw_1d_c3d4_xpl_p_pp.inp	C3D4 elements.
iw_1d_c3d5_xpl_p_pp.inp	C3D5 elements.
iw_1d_c3d6_xpl_p_pp.inp	C3D6 elements.
iw_1d_c3d8r_xpl_p_pp.inp	C3D8R elements.
iw_1d_c3d10m_xpl_p_pp.inp	C3D10M elements.
iw_1d_cax3_xpl_p_pp.inp	CAX3 elements.
iw_1d_cax4r_xpl_p_pp.inp	CAX4R elements.
iwt_1d_c3d6_xpl_p_pp.inp	C3D6 elements.
iwt_1d_cax3_xpl_p_pp.inp	CAX3 elements.

Spherical wavefront:

iw_1d_cpe3_xpl_s_pp.inp	CPE3 elements.
iw_1d_cpe4r_xpl_s_pp.inp	CPE4R elements.
iw_1d_cpe6m_xpl_s_pp.inp	CPE6M elements.
iw_1d_cps3_xpl_s_pp.inp	CPS3 elements.
iw_1d_cps4r_xpl_s_pp.inp	CPS4R elements.
iw_1d_c3d4_xpl_s_pp.inp	C3D4 elements.
iw_1d_c3d5_xpl_s_pp.inp	C3D5 elements.
iw_1d_c3d6_xpl_s_pp.inp	C3D6 elements.
iw_1d_c3d8r_xpl_s_pp.inp	C3D8R elements.
iw_1d_c3d10m_xpl_s_pp.inp	C3D10M elements.
iw_1d_cax3_xpl_s_pp.inp	CAX3 elements.
iw_1d_cax4r_xpl_s_pp.inp	CAX4R elements.
iwt_1d_cpe4r_xpl_s_pp.inp	CPE4R elements.
iwt_1d_c3d8r_xpl_s_pp.inp	C3D8R elements.

Bubble-loading amplitude:

iw_1d_cpe4r_xpl_b_pp.inp	CPE4R elements.

Coupled tests

Elements tested

AC2D3 AC2D4 AC2D4R AC2D6 AC2D8 AC3D6 AC3D8 AC3D8R
ACAX3 ACAX4 ACAX6
B21 B21H B22 B22H B23
S3R S4R S4RS STRI3 SAX1 SAX2
C3D6 CAX3 CPE4R CPE6M CPEG4R CPS4R CPS8R

Features tested

Incident wave loading in Abaqus/Standard and Abaqus/Explicit with solid-fluid coupling using a surface-based tie constraint.

Problem description

One-dimensional incident wave loading is tested for coupled analysis in this verification set. When solid and beam elements are coupled with the acoustic elements, the sound source is located at (−10, 0, 0) for the planar waves and at (−100000, 0, 0) for the spherical waves. For coupling with shell elements the sound source is located at (0, 0, 10) for the planar waves and at (0, 0, 100000) for the spherical waves. For all the axisymmetric cases the sound source is located at (0, −10) for the planar waves and at (0, −100000) for the spherical waves. The standoff point is located at (0, 0, 0).

One acoustic element is used for the coupling analysis. The two-dimensional acoustic element has a length and width of 1 m and a thickness of 10^{-4} m. The three-dimensional acoustic element has unit length on all sides. The material properties for the acoustic elements are as follows: density, $\rho_f = 1.21$ kg/m^3; bulk modulus, $K_f = 1.424 \times 10^5$ Pa. The material properties of the surrounding medium are the same as those of the fluid. The planar shells are modeled in the *X*–*Y* plane with a surface lying on one face of the acoustic element. The shell element thickness is 10^{-4} m. The beam elements are modeled parallel to the *y*-direction and lying on one edge of the two-dimensional acoustic element. The beam has a square cross-section area of 10^{-4} m^2. Solid elements are modeled with the length direction as the *x*-axis and the other two directions parallel to the *y*- and *z*-axes; they

are placed adjacent to the acoustic elements. In axisymmetric cases the elements are oriented in the axial direction. The material properties of the solid and structural elements are the same as those used in the previous cases.

All nodes are kept fixed for the beam and shell elements. For the solid elements all nodes are fixed in the y-direction, and the nodes that are further away from the tied surface are fixed additionally in the x-direction. For the acoustic elements the loading is applied as a ramp function attaining a maximum of 2.0755 Pa at the end of the step at 5 ms. Additionally, pressure is applied for the solid and structural elements as a ramp function with a maximum of 5 Pa at the end of the step. The results are compared with the expected values of reaction forces and POR.

Two similar models are also created to test the bubble loading, with water used as the material instead of air.

Results and discussion

The results exactly match the expected values for all cases.

Input files

Abaqus/Standard input files

Planar wavefront, pressure amplitude, and geometric nonlinearities are ignored:

iw_1d_sac_b_dyl_p_pa.inp	AC2D4/B23 elements.
iw_1d_sac_s_dyl_p_pa.inp	ACAX4/SAX1 elements.
iw_1d_sac_c_dyl_p_pa.inp	AC2D4/CPE4R elements.

Spherical wavefront, pressure amplitude, and geometric nonlinearities are ignored:

iw_1d_sac_b_dyl_s_pa.inp	AC2D8/B22H elements.
iw_1d_sac_s_dyl_s_pa.inp	AC3D8/STRI3 elements.
iw_1d_sac_c_dyl_s_pa.inp	AC3D8/C3D8 elements.

Planar wavefront, acceleration amplitude, and geometric nonlinearities are ignored:

iw_1d_sac_b_dyl_p_aa.inp	AC2D4/B23 elements.
iw_1d_sac_s_dyl_p_aa.inp	ACAX4/SAX1 elements.
iw_1d_sac_c_dyl_p_aa.inp	AC2D4/CPE4R elements.

Planar wavefront, pressure amplitude, and geometric nonlinearities are considered:

iw_1d_sac_b_dyn_p_pa.inp	AC2D8/B22 elements.
iw_1d_sac_s_dyn_p_pa.inp	AC3D8/S4R elements.
iw_1d_sac_c_dyn_p_pa.inp	AC2D3/CPEG4R elements.

Spherical wavefront, pressure amplitude, and geometric nonlinearities are considered:

iw_1d_sac_b_dyn_s_pa.inp	AC2D4/B21H elements.
iw_1d_sac_s_dyn_s_pa.inp	ACAX6/SAX2 elements.
iw_1d_sac_c_dyn_s_pa.inp	AC2D6/CPE6M elements.

Planar wavefront, acceleration amplitude, and geometric nonlinearities are considered:

iw_1d_sac_b_dyn_p_aa.inp	AC2D8/B22 elements.
iw_1d_sac_s_dyn_p_aa.inp	AC3D8/S4R elements.
iw_1d_sac_c_dyn_p_aa.inp	AC2D3/CPEG4R elements.

Bubble-loading amplitude:

iw_1d_sac_b_dyl_b_pa.inp	AC2D8/B22H elements.
iw_1d_sac_c_dyl_b_pa.inp	AC3D8/C3D8 elements.

Abaqus/Explicit input files

Planar wavefront, pressure amplitude:

iw_1d_sac_b_xpl_p_pa.inp	AC2D3/B21 elements.
iw_1d_sac_s_xpl_p_pa.inp	AC3D8R/S3R elements.
iw_1d_sac_c_xpl_p_pa.inp	AC2D4/CPS4R elements.

Spherical wavefront, pressure amplitude:

iw_1d_sac_b_xpl_s_pa.inp	AC2D4R/B21 elements.
iw_1d_sac_s_xpl_s_pa.inp	AC3D8R/S4RS elements.
iw_1d_sac_c_xpl_s_pa.inp	ACAX3/CAX3 elements.

Planar wavefront, acceleration amplitude:

iw_1d_sac_b_xpl_p_aa.inp	AC2D3/B21 elements.
iw_1d_sac_s_xpl_p_aa.inp	AC3D8R/S3R elements.
iw_1d_sac_c_xpl_p_aa.inp	AC2D4/CPS4R.

Incident wave reflection: spherical waves

Elements tested

S4R AC3D8 AC3D8R

Features tested

Incident wave reflection in Abaqus/Standard and Abaqus/Explicit with solid-fluid coupling.

Problem description

These are single-element tests that model a sound source at (0.0, 0.0, 10.0) for the spherical waves and a reflecting surface 5 m directly above the sound source. The standoff point is located at (0.0, 0.0, 0.0). The planar shell is modeled in the X–Y plane with unit length on all sides. The shell thickness is 10^{-4} m. All nodes are fixed for the planar shells. The shell material properties are as follows: E=10^6 Pa and ρ=1000 kg/m^3. The three-dimensional acoustic element is modeled with one face of the element on the X–Y plane and has unit length on all sides. The material properties are the same as those used in the previous case. The surrounding medium has the following material properties: density, ρ_f=100 kg/m^3; bulk modulus, K_f=10^8 Pa. The loading is a step function with pressure magnitude of 1000 Pa for planar shells and 415.09517 Pa for acoustic elements. Four different properties of the reflecting surface are considered for the tests. For planar shells the reaction forces are compared with the expected values. For acoustic elements POR values are compared.

Results and discussion

The results exactly match the expected values for all cases.

Input files

Abaqus/Standard input files

iwr_1d_1_dyl_s_pp.inp	S4R element with $1/c_1$=0.

iwr_1d_2_dyl_s_pp.inp	S4R element with $1/c_1=1/\rho_f c$.
iwr_1d_3_dyl_s_pp.inp	S4R element with $1/c_1 >> 1/\rho_f c$.
iwr_1d_4_dyl_s_pp.inp	S4R element with $1/c_1=0.5$.
iwr_1d_1_dyl_s_pa.inp	AC3D8 element with $1/c_1=0$.
iwr_1d_2_dyl_s_pa.inp	AC3D8 element with $1/c_1=1/\rho_f c$.
iwr_1d_3_dyl_s_pa.inp	AC3D8 element with $1/c_1 >> 1/\rho_f c$.
iwr_1d_4_dyl_s_pa.inp	AC3D8 element with $1/c_1=0.5$.

Abaqus/Explicit input files

iwr_1d_1_xpl_s_pp.inp	S4R element with $1/c_1=0$.
iwr_1d_2_xpl_s_pp.inp	S4R element with $1/c_1=1/\rho_f c$.
iwr_1d_3_xpl_s_pp.inp	S4R element with $1/c_1 >> 1/\rho_f c$.
iwr_1d_4_xpl_s_pp.inp	S4R element with $1/c_1=0.5$.
iwtr_1d_1_xpl_s_pp.inp	S4R element with $1/c_1=0$.
iwr_1d_1_xpl_s_pa.inp	AC3D8R element with $1/c_1=0$.
iwr_1d_2_xpl_s_pa.inp	AC3D8R element with $1/c_1=1/\rho_f c$.
iwr_1d_3_xpl_s_pa.inp	AC3D8R element with $1/c_1 >> 1/\rho_f c$.
iwr_1d_4_xpl_s_pa.inp	AC3D8R element with $1/c_1=0.5$.
iwtr_1d_1_xpl_s_pa.inp	AC3D8R element with $1/c_1=0$.
iwtr_1d_3_xpl_s_pa.inp	AC3D8R element with $1/c_1 >> 1/\rho_f c$.

Incident wave reflection: planar waves

Elements tested

S4R AC3D8 AC3D8R

Features tested

Incident wave reflection in Abaqus/Standard and Abaqus/Explicit.

Problem description

These are single-element tests that model a sound source at (0.0, 10.0, 10.0) for the direct-path waves and a reflecting surface 20 m directly below the sound source. The standoff point is located at (0.0, 0.0, 0.0). The loading amplitude is a step function with pressure magnitude of 1000 Pa for the planar shells and 1.0 Pa for the acoustic elements.

The planar shell is modeled in the *X*–*Y* plane with unit length on all sides. The shell thickness is 10^{-4} m. All nodes are fixed for the planar shells. The shell material properties are as follows: $E=10^6$ Pa and $\rho=1000$ kg/m^3.

The three-dimensional acoustic element is modeled with one face of the element on the *X*–*Y* plane and has unit length on all sides. The acoustic medium has the following material properties: density, $\rho_f=1.0$ kg/m^3; bulk modulus, $K_f=1.6 \times 10^5$ Pa, resulting in a speed of sound of 400 m/s.

For planar shells the reaction forces are compared with the expected values. For acoustic elements POR values are compared.

Results and discussion

The results exactly match the expected values for all cases.

Input files

Abaqus/Standard input files

iwr_1d_1_dyl_p_pp.inp	S4R element with $1/c_1=0$.
iwr_1d_1_dyl_p_pa.inp	AC3D8 element with $1/c_1=0$.

Abaqus/Explicit input files

iwr_1d_1_xpl_p_pp.inp	S4R element with $1/c_1=0$.
iwtr_1d_1_xpl_p_pp.inp	S4R element with $1/c_1=0$, total wave formulation.
iwr_1d_1_xpl_p_pa.inp	AC3D8R element with $1/c_1=0$.
iwtr_1d_1_xpl_p_pa.inp	AC3D8R element with $1/c_1=0$, total wave formulation.

Incident wave interaction in steady-state dynamics

Elements tested

S4R C3D8 AC3D8 AC3D8R

Features tested

Incident wave interaction in Abaqus/Standard.

Problem description

These are simple tests to verify the application of planar, spherical, and diffuse incident wave fields in steady-state dynamics.

Results and discussion

The results match the expected values for all cases.

Input files

Abaqus/Standard input files

ac3d8_iwissdd.inp	AC3D8 element using the direct-solution steady-state dynamic procedure.
ac3d8_iwissdd2.inp	AC3D8 element using the direct-solution steady-state dynamic procedure.
ac3d8_iwissdd_lc.inp	AC3D8 element using the direct-solution steady-state dynamic procedure and the **LOAD CASE* option.
ac3d8_iwissds.inp	AC3D8 element using the subspace-based steady-state dynamic procedure.
c3d8_iwissdd.inp	C3D8 element using the direct-solution steady-state dynamic procedure.
c3d8_iwissdd2.inp	C3D8 element using the direct-solution steady-state dynamic procedure.

c3d8_iwissdd_lc.inp	C3D8 element using the direct-solution steady-state dynamic procedure and the **LOAD CASE* option.
c3d8_iwissds.inp	C3D8 element using the subspace-based steady-state dynamic procedure.
stl_case1.inp	AC3D8 and S4R elements.
stl_case2.inp	AC3D8 and S4R elements.
iwi_diffuse_s4.inp	S4R element with diffuse loading.

Distributed traction and edge loads

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Features tested

This section provides basic verification tests for the traction load labels TRVEC and TRSHR and the edge load labels EDLD, EDNOR, EDSHR, and EDTRA using the distributed element-based and surface loads.

Distributed shear and general traction loads

Elements tested

CPS3 CPE3 CPS4 CPE4 CPS6 CPE6 CPS6M CPE6M CPS8 CPE8
CPEG3 CPEG4 CPEG6 CPEG6M CPEG8
CAX3 CAX4 CAX6 CAX6M CAX8
CGAX3 CGAX4 CGAX6 CGAX6M CGAX8
C3D4 C3D5 C3D8R C3D6 C3D10 C3D10M C3D15 C3D20 C3D27
CSS8
CCL9 CCL12 CCL18 CCL24
S3R STRI3 S4R S4R5 STRI65 S8R S8R5 S9R5
SC6R SC8R
SAX1 SAX2 RAX2
M3D3 M3D4 M3D6 M3D8 M3D9
MAX1 MAX2 MGAX1 MGAX2
MCL6 MCL9
SFMCL6 SFMCL9
SFM3D3 SFM3D4 SFM3D6 SFM3D8
SFMAX1 SFMAX2 SFMGAX1 SFMGAX2
R2D2 R3D3 R3D4 RAX2

Problem description

The analyses in this section test the traction load labels TRVEC and TRSHR using distributed element-based and surface loads. One-element and two-element tests are performed to verify the loading options on all the faces of supported elements. In both Abaqus/Standard and Abaqus/Explicit tests, the elements are held fixed by kinematic coupling constraints as each face of each element is loaded with a combination of distributed general tractions and shear tractions. The resultant forces at the kinematic reference nodes are output to verify that distributed loads are properly applied to each element.

Results and discussion

The results for each combination indicate that the loads are applied correctly.

Input files

Abaqus/Standard input files

tracload2d.inp	Traction loading of two-dimensional elements.
tracloadcpeg.inp	Traction loading of generalized plane strain elements.
tracloadcax.inp	Traction loading of axisymmetric elements.
tracloadcgax.inp	Traction loading of axisymmetric elements with twist.
tracload3d.inp	Traction loading of three-dimensional elements.
tracloadcss8.inp	Traction loading of continuum solid shell elements.
tracloadccl.inp	Traction loading of cylindrical elements.
tracloadshl.inp	Traction loading of shell elements.
tracloadsc.inp	Traction loading of continuum shell elements.
tracloadrsax.inp	Traction loading of axisymmetric shell elements and axisymmetric rigid link elements.
tracloadm3d.inp	Traction loading of three-dimensional membrane and surface elements.
tracloadmax.inp	Traction loading of axisymmetric membrane elements.
tracloadmgax.inp	Traction loading of axisymmetric membrane elements with twist.
tracloadmcl.inp	Traction loading of cylindrical membrane elements.
tracloadsfmax.inp	Traction loading of axisymmetric surface elements.
tracloadsfmgax.inp	Traction loading of axisymmetric surface elements with twist.
tracloadr2d.inp	Traction loading of two-dimensional rigid elements.
tracloadr3d.inp	Traction loading of three-dimensional rigid elements.

Abaqus/Explicit input files

tracload2d_xpl.inp	Traction loading of two-dimensional elements.
tracloadcax_xpl.inp	Traction loading of axisymmetric elements.
tracload3d_xpl.inp	Traction loading of three-dimensional elements.
tracloadshl_xpl.inp	Traction loading of shell, membrane, and surface elements.
tracloadsc_xpl.inp	Traction loading of continuum shell elements.
tracloadrsax_xpl.inp	Traction loading of axisymmetric shell elements and axisymmetric rigid link elements.
tracloadr2d2_xpl.inp	Traction loading of two-dimensional rigid elements.

Distributed edge loads

Elements tested

S3R STRI3 S4R S4R5 STRI65 S8R S8R5 S9R5

Problem description

The analyses in this section test the edge load labels EDLD, EDNOR, EDSHR, and EDTRA using distributed element-based and surface loads. One-element and two-element tests are performed to verify the loading options on all the edges of supported shell elements. In both Abaqus/Standard and Abaqus/Explicit tests, the elements are held fixed by kinematic coupling constraints as each edge of each element is loaded with a combination of

distributed edge loads. The resultant forces at the kinematic reference nodes are output to verify that distributed loads are properly applied to each element.

Results and discussion

The results for each combination indicate that the loads are applied correctly.

Input files

Abaqus/Standard input file

tracloadedge.inp	Edge loading of shell elements.

Abaqus/Explicit input file

tracloadedge_xpl.inp	Edge loading of shell elements.

Distributed shear and general traction loads in geometrically nonlinear analyses

Elements tested

CPS3 CPE3 CPS4 CPE4 CPS6 CPE6 CPS6M CPE6M CPS8 CPE8
C3D4 C3D5 C3D8R C3D6 C3D10 C3D10M C3D15 C3D20
CCL9 CCL12 CCL18 CCL24
S3R STRI3 S4R S4R5 STRI65 S8R S8R5 S9R5
SC6R SC8R
SAX1 SAX2

Problem description

The analyses in this section test the traction load labels TRVEC and TRSHR using distributed element-based and surface loads in geometrically nonlinear analyses. Tests include models under large rigid body rotations and large deformations. In the tests where elements undergo large rigid body rotations, one facet is coupled to a kinematic coupling reference node. A traction load is applied to another face. This load is kept constant as the elements are rotated by the kinematic coupling reference node. The reaction forces at the kinematic reference node are used to verify that the loads are properly applied and rotated with the element. Different combinations of the follower and non-follower surface loads and constant resultants are also used. Some of the models in the tests have cylindrical geometry. General traction or shear loadings are applied on the cylindrical surface by defining a local cylindrical coordinate system.

Results and discussion

The results for each combination indicate that the loads are applied correctly.

Input files

Abaqus/Standard input files

traclarge_rotation_2d.inp	Traction loading of two-dimensional elements.
traclarge_rotation_3d.inp	Traction loading of three-dimensional elements.
traclarge_rotation_3d_usub.inp	User-defined traction loading of three-dimensional elements.
traclarge_rotation_3d_usub.f	User subroutine used in traclarge_rotation_3d_usub.inp.
traclarge_rotation_shl.inp	Traction loading of three-dimensional shell elements.
traclarge_rotation_m3d.inp	Traction loading of three-dimensional membrane elements.

tracnlgeom_ccl9.inp	Traction loading of 9-node cylindrical element CCL9.
tracnlgeom_ccl12.inp	Traction loading of 12-node cylindrical element CCL12.
tracnlgeom_ccl12_usub.inp	User-defined traction loading of 12-node cylindrical element CCL12.
tracnlgeom_ccl12_usub.f	User subroutine used in tracnlgeom_ccl12_usub.inp.
tracnlgeom_ccl18.inp	Traction loading of 18-node cylindrical element CCL18.
tracnlgeom_ccl24.inp	Traction loading of 24-node cylindrical element CCL24.
tracnlgeom_sax.inp	Traction loading of axisymmetric shell element.
trac_cylori.inp	Traction loading of a three-dimensional cylinder.

Abaqus/Explicit input files

traclarge_rotation_2d_xpl.inp	Traction loading of two-dimensional elements.
trac_cylori_xpl.inp	Traction loading of a three-dimensional cylinder.

Distributed edge loads in a geometrically nonlinear analysis

Elements tested

S3R STRI3 S4R S4R5 STRI65 S8R S8R5 S9R5

Problem description

The analyses in this section test the edge load labels EDLD, EDNOR, EDSHR, and EDTRA using distributed element-based and surface loads in geometrically nonlinear analyses. One facet is coupled to a kinematic coupling reference node. A traction load is applied to another face. This load is kept constant as the elements are rotated by the kinematic coupling reference node. The reaction forces at the kinematic reference node are used to verify that the loads are properly applied and rotated with the element. Different combinations of the follower and non-follower surface loads and constant resultants are also used.

Results and discussion

The results for each combination indicate that the loads are applied correctly.

Input files

Abaqus/Standard input files

tracedgelarge_rotation.inp	Edge loading of shell elements.
tracnlgeom_edge_usub.inp	User-defined edge loading of shell elements.
tracnlgeom_edge_usub.f	User subroutine used in tracnlgeom_edge_usub.inp.

Abaqus/Explicit input file

traclarge_rotation_edge_xpl.inp	Edge loading of shell elements.

Dead load analysis of a membrane structure using a constant resultant

Elements tested

M3D4

Problem description

This section provides basic verification of using constant resultants in a dead load analysis. The constant resultant method has certain advantages when a traction is used to model a distributed load with a known constant resultant.

If you choose not to have a constant resultant, the traction vector is integrated over the surface in the current configuration, a surface that in general deforms in a geometrically nonlinear analysis. The most common example of a traction that should be integrated over the current configuration is a live pressure load defined as $\mathbf{t} = -p\mathbf{n}$, where $\mathbf{n}$ is the normal in the current configuration. The total resultant due to a pressure load depends on the surface area in the current configuration. A live uniform normal surface traction integrated over the current surface is equivalent to applying a uniform pressure load. By default, the traction vector is integrated over the surface in the current configuration.

If you choose to have a constant resultant, the traction vector is integrated over the surface in the reference configuration, which is constant.

The analysis in this section consists of a unit planar membrane structure that is held fixed at the edges by a kinematic coupling constraint. The normal of the flat structure is in the $(1, 0, 1)$ direction. A uniform dead traction load (of magnitude 4) is applied in the negative $\mathbf{e_3}$-direction. This could be considered a simple model of a sloped roof with a snow load.

Let $S_o = 1$ and S denote the total surface area of the plate in the reference and current configurations, respectively. With no constant resultant, the total integrated load on the plate, $\mathbf{f}$, is

$$\mathbf{f} = \int_S \mathbf{t}\, dS = \int_S -4\,\mathbf{e}_2\, dS = -4\,\mathbf{e}_3\, S.$$

In this case a uniform traction leads to a resultant load that increases as the surface area of the plate increases, which is not consistent with a fixed snow load. With the constant resultant method, the total integrated load on the plate is

$$\mathbf{f} = \int_{S_o} \mathbf{t}\, dS_o = \int_{S_o} -4\,\mathbf{e}_3\, dS_o = -4\,\mathbf{e}_3\, S_o = -4\,\mathbf{e}_3\,.$$

In the first step the load is applied without a constant resultant. In the second step the structure is unloaded. In the third step the load is applied with a constant resultant.

Results and discussion

The magnitude of the reaction force at the kinematic coupling reference node at the end of the first step is 4.59. A reaction force greater than 4.0 reflects the fact that the surface area of the membrane is increasing with the load. The magnitude of the reaction force at the kinematic coupling reference node at the end of the third step is 4.0 as expected.

Input files

Abaqus/Standard input files

tracresultant_m3d4.inp	Testing the CONSTANT RESULTANT parameter.
tracresultant_m3d4_usub.inp	User-defined traction loading with the CONSTANT RESULTANT parameter.
tracresultant_m3d4_usub.f	User subroutine used in tracresultant_m3d4_usub.inp.

Abaqus/Explicit input file

tracresultant_m3d4_xpl.inp	Testing the CONSTANT RESULTANT parameter.

Patch tests

In this section:

- *Membrane patch test*
- *Patch test for three-dimensional solid elements*
- *Patch test for cylindrical elements*
- *Patch test for axisymmetric elements*
- *Patch test for axisymmetric elements with twist*
- *Patch test for plate bending*
- *Patch test for beam elements*
- *Patch test for heat transfer elements*
- *Patch test for thermal-electrical elements*
- *Patch test for acoustic elements*

Membrane patch test

Products: Abaqus/Standard Abaqus/Explicit

Overview

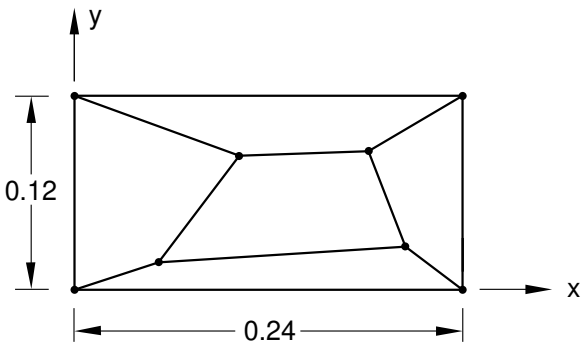
This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

CPE3 CPE3H CPE3T CPE4 CPE4H CPE4I CPE4IH CPE4R CPE4RH CPE4RHT CPE4RT

CPE6 CPE6H CPE6M CPE6MH CPE6MHT CPE6MT CPE8 CPE8H CPE8R CPE8RH
CPEG3 CPEG3H CPEG4 CPEG4H CPEG4I CPEG4IH CPEG4R CPEG4RH
CPEG6 CPEG6H CPEG6M CPEG6MH CPEG8 CPEG8H CPEG8R CPEG8RH
CPS3 CPS3T CPS4 CPS4I CPS4R CPS4RT CPS6 CPS6M CPS6MT CPS8 CPS8R
M3D3 M3D4 M3D4R M3D6 M3D8 M3D8R M3D9 M3D9R
S3 S3R S4 S4R S4R5 S8R S8R5 S9R5 STRI3 STRI65
CSS8

Problem description



Model: Thickness, t=0.001.

Material: Linear elastic, Young's modulus = 1.0×10^6, Poisson's ratio = 0.25.

For the coupled temperature-displacement elements dummy thermal properties are prescribed to complete the material definition.

Loading/boundary conditions for Step 1: $u_x = 10^{-3}((x + y/2)$, $u_y = 10^{-3}(y + x/2)$ at all exterior nodes. For shell elements, $u_z = 0$ at all nodes.

In the Abaqus/Explicit simulations this step is followed by an intermediate step in which the model is returned to its unloaded state.

Loading/boundary conditions for Step 2: Rigid body motion is constrained. Uniform edge pressure = 10000.

Loading/boundary conditions for Step 3: $u_x = 10^{-3}(x + y/2)$, $u_y = 10^{-3}(y + x/2)$ at all exterior nodes, where x and y are the nodal coordinates of the undeformed geometry. For shell elements $u_z = 0$ at all nodes.

In the Abaqus/Standard simulations this step is defined as a perturbation step; in the Abaqus/Explicit simulations a velocity boundary condition that gives rise to the perturbation is specified instead.

Reference solution

The analytical results for each step are presented below.

Step 1: PERTURBATION

- $\sigma_x = \sigma_y =$ 1333 for plane stress, shell, and membrane elements.
- $\sigma_x = \sigma_y =$ 1600 for plane strain elements.
- $\sigma_z =$ 800 for plane strain elements.
- $\sigma_{xy} =$ 400 for all elements.
- $\varepsilon_x = \varepsilon_y = \gamma_{xy} = 10^{-3}$.

Step 2: NLGEOM

Element	Strain	Edge	σ_x, σ_y	ϵ_x, ϵ_y
Category	Measure	Thickness		(10^{-3})
Plane strain	Log	Original	10000	6.25
Plane stress	Log	New	10153	7.62
Membrane	Log	New	10076	7.56
Shell	Green's	Original	9926	7.44
F.S. Shells	Log	New	10076	7.56

The hand-calculated solutions will differ because of the various assumptions made for each category of element. The assumptions made correspond to those that are implemented in Abaqus. The two that cause significant differences in the results of this step are the strain measure used and the elemental cross-sectional area used to calculate the edge load and output stresses.

The strain measure used for shells, for example, is Green's strain. This strain measure is intended for large displacements and rotations but small strains. The remainder of the elements, including finite-strain shells, use logarithmic strain, which is intended for large-strain analyses.

The use of nonlinear geometric effects implies that the nodal coordinates will change for each element. This, in turn, implies that the cross-sectional area of the elements will change. The change of length and width is taken into account for all elements. This is not the case for the thickness, however. The thickness of the plane strain elements, of course, is assumed to remain constant. The thickness is also assumed to remain constant for the shell elements, excluding finite-strain shells. The remainder of the elements take into account a change in thickness determined by assuming constant elemental volume. This change in thickness, combined with a change in length

and width, results in a cross-sectional area that differs from the initial area. This result affects the output stress calculations, as well as the applied edge load.

Since the edge load is calculated as the pressure divided by the area, the edge load will vary because of the variation in the cross-sectional area. Edge loads are presently not available for shells and membranes. Equivalent concentrated nodal forces are applied to these elements in this step, and as a result the load remains constant.

In the Abaqus/Explicit simulations this is the third step. (The second step in the Abaqus/Explicit simulations returns the model to its unloaded state.)

Step 3: PERTURBATION

- $\sigma_x = \sigma_y =$ 1323 for plane stress, shell, and membrane elements.
- $\sigma_x = \sigma_y =$ 1590 for plane strain elements.
- $\sigma_z =$ 795 for plane strain elements.
- $\sigma_{xy} =$ 397.0 for plane stress, shell, and membrane elements.
- $\sigma_{xy} =$ 397.5 for plane strain elements.
- $\varepsilon_x = \varepsilon_y = \gamma_{xy} = 9.92 \times 10^{-3}$ for plane stress, shell, and membrane elements.
- $\varepsilon_x = \varepsilon_y = \gamma_{xy} = 9.94 \times 10^{-3}$ for plane strain elements.

  In the Abaqus/Explicit simulations this is the fourth step. The results from the third step in the Abaqus/Explicit simulations must be subtracted from the results of the fourth step to obtain the perturbation about the loaded state.

Results and discussion

All elements yield exact solutions except for the three-dimensional shells (other than the finite-strain shells), which differ from the analytical solution by about 2%. These elements are recommended only for analyses with large displacements and/or large rotations and small strains. The finite-strain shells are recommended for analyses that experience large strains.

To obtain the exact solution, the patch tests of the CPEG3, CPEG4, and CPEG4I elements require a convergence tolerance that is tighter than the default. The necessary tolerance is set with the solution controls.

These tests also verify the specification of a nondefault thickness for plane stress elements and membrane elements. The strain energy, which is dependent on the element thickness, was calculated from the previously verified values of the stress and strain and successfully compared to the Abaqus variable ALLIE. This result indicates that the nondefault thickness is being used correctly.

Section output requests to the results (`.fil`) file and to the data (`.dat`) file are used in some of the input files with CPE3, CPE8H, and CPEG4RH elements to output accumulated quantities in different sections through the model.

Input files

Abaqus/Standard input files

ece3sfp1.inp	CPE3 elements.
ece3shp1.inp	CPE3H elements.
ece4sfp1.inp	CPE4 elements.
ece4shp1.inp	CPE4H elements.

ece4sip1.inp	CPE4I elements.
ece4sjp1.inp	CPE4IH elements.
ece4srp1.inp	CPE4R elements.
ece4syp1.inp	CPE4RH elements.
ece4typ1.inp	CPE4RHT elements.
ece4trp1.inp	CPE4RT elements.
ece6sfp1.inp	CPE6 elements.
ece6shp1.inp	CPE6H elements.
ece6skp1.inp	CPE6M elements.
ece6slp1.inp	CPE6MH elements.
ece6tlp1.inp	CPE6MHT elements.
ece6tkp1.inp	CPE6MT elements.
ece8sfp1.inp	CPE8 elements.
ece8shp1.inp	CPE8H elements.
ece8srp1.inp	CPE8R elements.
ece8syp1.inp	CPE8RH elements.
ecg3sfp1.inp	CPEG3 elements.
ecg3shp1.inp	CPEG3H elements.
ecg4sfp1.inp	CPEG4 elements.
ecg4shp1.inp	CPEG4H elements.
ecg4sip1.inp	CPEG4I elements.
ecg4sjp1.inp	CPEG4IH elements.
ecg4srp1.inp	CPEG4R elements.
ecg4syp1.inp	CPEG4RH elements.
ecg6sfp1.inp	CPEG6 elements.
ecg6shp1.inp	CPEG6H elements.
ecg6skp1.inp	CPEG6M elements.
ecg6slp1.inp	CPEG6MH elements.
ecg8sfp1.inp	CPEG8 elements.
ecg8shp1.inp	CPEG8H elements.
ecg8srp1.inp	CPEG8R elements.
ecg8syp1.inp	CPEG8RH elements.
ecs3sfp1.inp	CPS3 elements.
ecs4sfp1.inp	CPS4 elements.
ecs4sfp1.f	User subroutine *DLOAD* used in ecs4sfp1.inp.
ecs4sip1.inp	CPS4I elements.
ecs4srp1.inp	CPS4R elements.
ecs6sfp1.inp	CPS6 elements.
ecs6skp1.inp	CPS6M elements.
ecs8sfp1.inp	CPS8 elements.
ecs8srp1.inp	CPS8R elements.
em33sfp1.inp	M3D3 elements.
em34sfp1.inp	M3D4 elements.

em34srp1.inp	M3D4R elements.
em36sfp1.inp	M3D6 elements.
em38sfp1.inp	M3D8 elements.
em38srp1.inp	M3D8R elements.
em39sfp1.inp	M3D9 elements.
em39srp1.inp	M3D9R elements.
esf3sxp1.inp	S3/S3R elements.
ese4sxp1.inp	S4 elements.
esf4sxp1.inp	S4R elements.
es54sxp1.inp	S4R5 elements.
es68sxp1.inp	S8R elements.
es58sxp1.inp	S8R5 elements.
es59sxp1.inp	S9R5 elements.
es63sxp1.inp	STRI3 elements.
es56sxp1.inp	STRI65 elements.
ess8fp1.inp	CSS8 elements.

Abaqus/Explicit input files

stresspatch_xpl_cpe3t.inp	CPE3T elements.
stresspatch_xpl_cpe4rt.inp	CPE4RT elements.
stresspatch_xpl_cpe6mt.inp	CPE6MT elements.
stresspatch_xpl_cps3t.inp	CPS3T elements.
stresspatch_xpl_cps4rt.inp	CPS4RT elements.
stresspatch_xpl_cps6mt.inp	CPS6MT elements.

Patch test for three-dimensional solid elements

Products: Abaqus/Standard Abaqus/Explicit

Overview

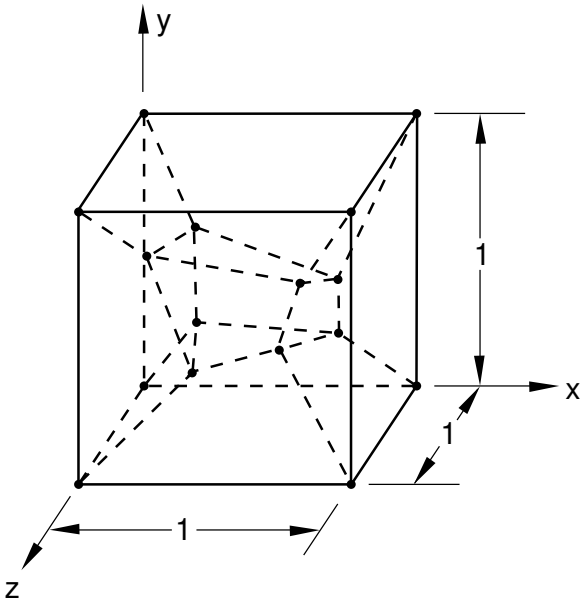
This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D4 C3D4H C3D4T C3D5 C3D6 C3D6H C3D6T
C3D8 C3D8H C3D8I C3D8IT C3D8IH C3D8IHT C3D8R C3D8RH C3D8RT C3D8T

C3D10 C3D10H C3D10HS C3D10M C3D10MH C3D10MHT C3D10MT
C3D15 C3D15H C3D15V C3D15VH
C3D20 C3D20H C3D20R C3D20RH C3D27 C3D27H C3D27R C3D27RH

Problem description



Material: Linear elastic, Young's modulus = 1.0×10^6, Poisson's ratio = 0.25.

For coupled temperature-displacement elements dummy thermal properties are prescribed to complete the material definition.

Loading for Step 1: Displacement boundary conditions at all exterior nodes: $u_x = 10^{-3}(2x + y + z)/2$, $u_y = 10^{-3}(x + 2y + z)/2$, $u_z = 10^{-3}(x + y + 2z)/2$.

In the Abaqus/Explicit simulations this step is followed by an intermediate step in which the model is returned to its unloaded state.

Loading for Step 2: Uniform pressure load: 10000. (Rigid body motion is constrained.)

Loading for Step 3: Displacement boundary conditions at all exterior nodes: $u_x = 10^{-3}(2x + y + z)/2$, $u_y = 10^{-3}(x + 2y + z)/2$, $u_z = 10^{-3}(x + y + 2z)/2$, where x, y, and z are the coordinates of the undeformed geometry.

In the Abaqus/Standard simulations this step is defined as a perturbation step; in the Abaqus/Explicit simulations a velocity boundary condition that gives rise to the perturbation is specified instead.

Reference solution

The analytical results for each step are presented below.

Step 1: PERTURBATION

- $\sigma_x = \sigma_y = \sigma_z = 2000.$
- $\sigma_{xy} = \sigma_{yz} = \sigma_{xz} = 400.$
- $\epsilon_x = \epsilon_y = \epsilon_z = 10^{-3}.$
- $\gamma_{xy} = \gamma_{yz} = \gamma_{xz} = 10^{-3}.$

Step 2: NLGEOM

- $\sigma_x = \sigma_y = \sigma_z = 10000.$
- $\sigma_{xy} = \sigma_{yz} = \sigma_{xz} = 0.$
- $\epsilon_x = \epsilon_y = \epsilon_z = 5.0 \times 10^{-3}.$
- $\gamma_{xy} = \gamma_{yz} = \gamma_{xz} = 0.$

In the Abaqus/Explicit simulations this is the third step. (The second step in the Abaqus/Explicit simulations returns the model to its unloaded state.)

Step 3: PERTURBATION

- $\sigma_x = \sigma_y = \sigma_z = 1990.$
- $\sigma_{xy} = \sigma_{yz} = \sigma_{xz} = 398.0.$
- $\epsilon_x = \epsilon_y = \epsilon_z = 9.95 \times 10^{-4}.$
- $\gamma_{xy} = \gamma_{yz} = \gamma_{xz} = 9.95 \times 10^{-4}.$

  In the Abaqus/Explicit simulations this is the fourth step. The results from the third step in the Abaqus/Explicit simulations must be subtracted from the results of the fourth step to obtain the perturbation about the loaded state.

Results and discussion

All elements except C3D27R and C3D27RH yield exact solutions. These elements use a special 14-point reduced-integration scheme since Gaussian $2 \times 2 \times 2$ integration leaves too many kinematic nodes. The stiffness

matrix is not integrated exactly with the employed integration rule, leading to small discrepancies in the results. The wedge elements and the quadratic reduced-integration brick elements pass only a restricted patch test; that is, such elements with midside nodes on any edges will pass the patch test only if those edges are straight.

Section output requests to the results (`.fil`) file and to the data (`.dat`) file are used in the input files with C3D8H, C3D10MH, and C3D27RH elements to output accumulated quantities in different sections through the model.

Input files

Abaqus/Standard input files

ec34sfp2.inp	C3D4 elements.
ec34shp2.inp	C3D4H elements.
ec35sfp2.inp	C3D5 elements.
ec36sfp2.inp	C3D6 elements.
ec36shp2.inp	C3D6H elements.
ec38sfp2.inp	C3D8 elements.
ec38shp2.inp	C3D8H elements.
ec38sip2.inp	C3D8I elements.
stresspatch_std_c3d8it.inp	C3D8IT elements.
ec38sjp2.inp	C3D8IH elements.
stresspatch_std_c3d8iht.inp	C3D8IHT elements.
ec38srp2.inp	C3D8R elements.
ec38syp2.inp	C3D8RH elements.
ec3asfp2.inp	C3D10 elements.
ec3ashp2.inp	C3D10H elements.
ec3asip2.inp	C3D10HS elements.
ec3askp2.inp	C3D10M elements.
ec3aslp2.inp	C3D10MH elements.
ec3atlp2.inp	C3D10MHT elements.
ec3atkp2.inp	C3D10MT elements.
ec3fsfp2.inp	C3D15 elements.
ec3fshp2.inp	C3D15H elements.
ec3isfp2.inp	C3D15V elements.
ec3ishp2.inp	C3D15VH elements.
ec3ksfp2.inp	C3D20 elements.
ec3kshp2.inp	C3D20H elements.
ec3ksrp2.inp	C3D20R elements.
ec3ksyp2.inp	C3D20RH elements.
ec3rsfp2.inp	C3D27 elements.
ec3rshp2.inp	C3D27H elements.
ec3rsrp2.inp	C3D27R elements.
ec3rsyp2.inp	C3D27RH elements.

Abaqus/Explicit input files

stresspatch_xpl_c3d4t.inp	C3D4T elements.
stresspatch_xpl_c3d6t.inp	C3D6T elements.
stresspatch_xpl_c3d8rt.inp	C3D8RT elements.
stresspatch_xpl_c3d8t.inp	C3D8T elements.

Patch test for cylindrical elements

Products: Abaqus/Standard

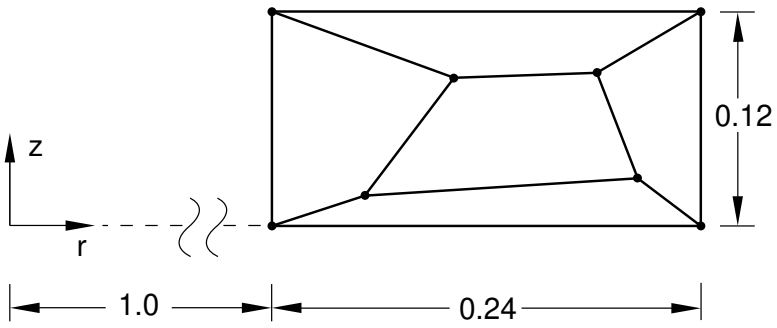
Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

CCL12 CCL24 CCL24R

Problem description



Material: Linear elastic, Young's modulus = 1.0×10^6, Poisson's ratio = 0.25.

Loading: Displacement boundary conditions applied to all exterior nodes: $u_r = 10^{-3}r, u_z = 10^{-3}(r+z)$.

Nonuniform body force: To maintain a constant shear stress $\sigma_{rz} = 400$ and preserve equilibrium, an equilibrating body force, BZNU, is defined in user subroutine *DLOAD* as BZNU$= -\sigma_{rz}/r = -400/r$, where r is the radius of the integration point.

Reference solution

The analytical results for each step are presented below.

Step : PERTURBATION

- $\sigma_r = \sigma_z = \sigma_\theta = 2000$.
- $\sigma_{rz} = 400$.
- $\epsilon_r = \epsilon_z = \epsilon_\theta = 10^{-3}$.
- $\gamma_{rz} = 10^{-3}$.

Results and discussion

All elements yield exact solutions.

Input files

Abaqus/Standard input files

ecccgfp1.inp	CCL12 elements.
ecccgfp1.f	User subroutine *DLOAD* used in ecccgfp1.inp.
eccrgfp1.inp	CCL24 elements.
eccrgfp1.f	User subroutine *DLOAD* used in eccrgfp1.inp.
eccrgrp1.inp	CCL24R elements.
eccrgrp1.f	User subroutine *DLOAD* used in eccrgrp1.inp.

Patch test for axisymmetric elements

Products: Abaqus/Standard Abaqus/Explicit

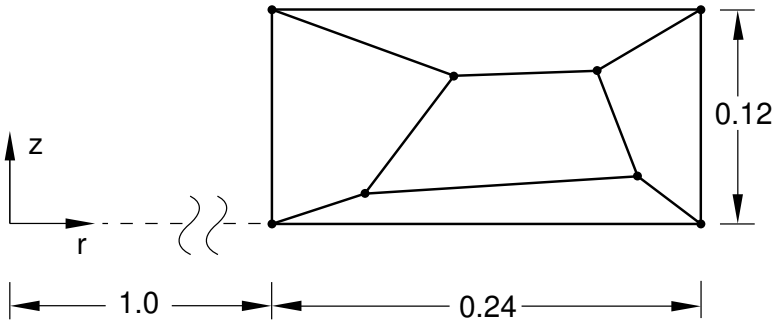
Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

CAX3 CAX3H CAX3T CAX4 CAX4H CAX4I CAX4IH CAX4R CAX4RH CAX4RHT CAX4RT
CAX6 CAX6H CAX6M CAX6MH CAX8 CAX8H CAX8R CAX8RH

Problem description



Material: Linear elastic, Young's modulus = 1.0×10^6, Poisson's ratio = 0.25.

For coupled temperature-displacement elements dummy thermal properties are prescribed to complete the material definition.

Loading for Step 1: Displacement boundary conditions applied to all exterior nodes: $u_r = 10^{-3}r$, $u_z = 10^{-3}(r + z)$.

Nonuniform body force: To maintain a constant shear stress $\sigma_{rz} = 400$ and preserve equilibrium, an equilibrating body force, BZNU, is defined in user subroutine *DLOAD* as BZNU$= -\sigma_{rz}/r = -400/r$, where r is the radius of the integration point.

In the Abaqus/Explicit simulations this step is followed by an intermediate step in which the model is returned to its unloaded state.

Loading for Step 2: Displacement boundary conditions applied to all exterior nodes: $u_r = 10^{-2}r$, $u_z = 10^{-2}z$.

Loading for Step 3: Displacement boundary conditions applied to the deformed geometry of Step 2 at all exterior nodes: $u_r = 10^{-3}r$, $u_z = 10^{-3}(r + z)$.

Nonuniform body force (as described for Step 1): BZNU$= -\sigma_{rz}/r = -400/r$.

In the Abaqus/Standard simulations this step is defined as a perturbation step; in the Abaqus/Explicit simulations a velocity boundary condition that gives rise to the perturbation is specified instead.

Reference solution

The analytical results for each step are presented below.

Step 1: PERTURBATION

- $\sigma_r = \sigma_z = \sigma_\theta =$ 2000.
- $\sigma_{rz} =$ 400.
- $\epsilon_r = \epsilon_z = \epsilon_\theta =$ 10^{-3}.
- $\gamma_{rz} =$ 10^{-3}.

Step 2: NLGEOM

- $\sigma_r = \sigma_z = \sigma_\theta =$ 19900.
- $\sigma_{rz} =$ 0
- $\epsilon_r = \epsilon_z = \epsilon_\theta =$ 9.95×10^{-3}.
- $\gamma_{rz} =$ 0.

In the Abaqus/Explicit simulations this is the third step. (The second step in the Abaqus/Explicit simulations returns the model to its unloaded state.)

Step 3: PERTURBATION

- $\sigma_r = \sigma_z = \sigma_\theta =$ 2000.
- $\sigma_{rz} =$ 400.
- $\epsilon_r = \epsilon_z = \epsilon_\theta =$ 1×10^{-3}.
- $\gamma_{rz} =$ 1×10^{-3}.

In the Abaqus/Explicit simulations this is the fourth step. The results from the third step in the Abaqus/Explicit simulations must be subtracted from the results of the fourth step to obtain the perturbation about the loaded state.

Results and discussion

All elements yield exact solutions.

Input files

Abaqus/Standard input files

eca3sfp5.inp	CAX3 elements.

eca3sfp5.f	User subroutine *DLOAD* used in eca3sfp5.inp.
eca3shp5.inp	CAX3H elements.
eca3shp5.f	User subroutine *DLOAD* used in eca3shp5.inp.
eca4sfp5.inp	CAX4 elements.
eca4sfp5.f	User subroutine *DLOAD* used in eca4sfp5.inp.
eca4shp5.inp	CAX4H elements.
eca4shp5.f	User subroutine *DLOAD* used in eca4shp5.inp.
eca4sip5.inp	CAX4I elements.
eca4sip5.f	User subroutine *DLOAD* used in eca4sip5.inp.
eca4sjp5.inp	CAX4IH elements.
eca4sjp5.f	User subroutine *DLOAD* used in eca4sjp5.inp.
eca4srp5.inp	CAX4R elements.
eca4srp5.f	User subroutine *DLOAD* used in eca4srp5.inp.
eca4syp5.inp	CAX4RH elements.
eca4syp5.f	User subroutine *DLOAD* used in eca4syp5.inp.
eca4typ5.inp	CAX4RHT elements.
eca4typ5.f	User subroutine *DLOAD* used in eca4typ5.inp.
eca4trp5.inp	CAX4RT elements.
eca4trp5.f	User subroutine *DLOAD* used in eca4trp5.inp.
eca6sfp5.inp	CAX6 elements.
eca6sfp5.f	User subroutine *DLOAD* used in eca6sfp5.inp.
eca6shp5.inp	CAX6H elements.
eca6shp5.f	User subroutine *DLOAD* used in eca6shp5.inp.
eca6skp5.inp	CAX6M elements.
eca6skp5.f	User subroutine *DLOAD* used in eca6skp5.inp.
eca6slp5.inp	CAX6MH elements.
eca6slp5.f	User subroutine *DLOAD* used in eca6slp5.inp.
eca8sfp5.inp	CAX8 elements.
eca8sfp5.f	User subroutine *DLOAD* used in eca8sfp5.inp.
eca8shp5.inp	CAX8H elements.
eca8shp5.f	User subroutine *DLOAD* used in eca8shp5.inp.
eca8srp5.inp	CAX8R elements.
eca8srp5.f	User subroutine *DLOAD* used in eca8srp5.inp.
eca8syp5.inp	CAX8RH elements.
eca8syp5.f	User subroutine *DLOAD* used in eca8syp5.inp.

Abaqus/Explicit input files

stresspatch_xpl_cax3t.inp	CAX3T elements.
stresspatch_xpl_cax4rt.inp	CAX4RT elements.
stresspatch_xpl_cax.f	User subroutine *VDLOAD* for the Abaqus/Explicit simulations.

Patch test for axisymmetric elements with twist

Products: Abaqus/Standard

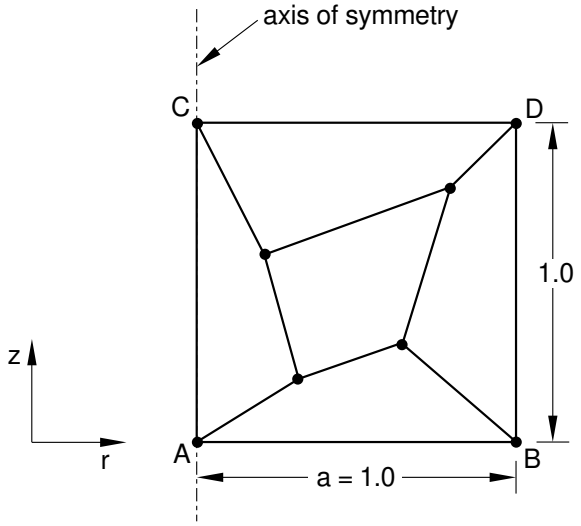
Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

CGAX3 CGAX3H CGAX3HT CGAX3T CGAX4 CGAX4H CGAX4HT CGAX4R CGAX4RH CGAX4T CGAX6 CGAX6H CGAX6M CGAX6MH CGAX8 CGAX8H CGAX8HT CGAX8R CGAX8RH CGAX8RHT CGAX8RT CGAX8T

Problem description



Material: Linear elastic, Young's modulus = 1.0×10^6, Poisson's ratio = 0.25, conductivity = 4.85×10^{-4}.

Loading for Step 1: A twist of 0.01 per unit length applied to face CD.

$\phi = 1.0 \times 10^{-2}$.

Loading for Step 2: Displacement boundary conditions applied to all exterior nodes: $u_r = 10^{-3}r$, $u_z = 10^{-3}(r + z)$, $\phi = 0$.

Nonuniform body force: To maintain a constant shear stress $\sigma_{rz} = 400$ and preserve equilibrium, an equilibrating body force, BZNU, is defined in user subroutine *DLOAD* as BZNU = $-\sigma_{rz}/r = -400/r$, where r is the radius of the integration point.

Loading for Step 3: Displacement boundary conditions applied to all exterior nodes: $u_r = 10^{-2}r$, $u_z = 10^{-2}z$, $\phi = 0$.

Loading for Step 4: Displacement boundary conditions applied to the deformed geometry of Step 2 at all exterior nodes: $u_r = 10^{-3}r$, $u_z = 10^{-3}(r+z)$, $\phi = 0$.

Nonuniform body force (as described for Step 2): BZNU = $-\sigma_{rz}/r = -400/r$.

Loading for Step 5: The displacement boundary conditions are the same as those applied in Step 3.

Temperatures are prescribed at every node along the boundary of the mesh. $T = C_1 r + C_2 z$, where T is the temperature, C_1 and C_2 are arbitrary constants, and r, z denote spatial location.

Nonuniform distributed flux: To maintain a uniform heat flux, q, a distributed heat flux, BFNU, is defined in user subroutine *DFLUX* as BFNU = $-C_1 k/r$, where r is the radius of the integration point and k is the conductivity.

Reference solution

The analytical results for each step are presented below.

Step 1: perturbation

Shear stress, $\sigma_{23} = G\phi r$, where r is the radial distance from the axis of symmetry and G is the shear modulus.

Resultant moment, $M = G\phi\pi a^4/2$ = 6283.2.

Step 2: perturbation

- $\sigma_r = \sigma_z = \sigma_\theta = 2000$.
- $\sigma_{rz} = 400$.
- $\epsilon_r = \epsilon_z = \epsilon_\theta = 10^{-3}$.
- $\gamma_{rz} = 10^{-3}$.

Step 3: geometrically nonlinear

- $\sigma_r = \sigma_z = \sigma_\theta = 19900$.
- $\sigma_{rz} = 0$.
- $\epsilon_r = \epsilon_z = \epsilon_\theta = 9.95 \times 10^{-3}$.
- $\gamma_{rz} = 0$.

Step 4: perturbation

- $\sigma_r = \sigma_z = \sigma_\theta = 2000$.
- $\sigma_{rz} = 400$.
- $\epsilon_r = \epsilon_z = \epsilon_\theta = 1 \times 10^{-3}$.

- $\gamma_{rz} = 1 \times 10^{-3}$.

Step 5: fully coupled thermal-stress

This step is applied only in tests of coupled temperature-displacement elements (CGAX*x*T).

Stresses and strains are the same as in Step 3. $q_r = -C_1 k$; $q_z = -C_2 k$.

Results and discussion

The results agree well with the analytical solution for all elements.

Section output requests to the results (`.fil`) file and to the data (`.dat`) file are used in the input files with CGAX8RH elements to output accumulated quantities in different sections through the model.

Input files

eca3gfp5.inp	CGAX3 elements.
eca3gfp5.f	User subroutine *DLOAD* used in eca3gfp5.inp.
eca3ghp5.inp	CGAX3H elements.
eca3ghp5.f	User subroutine *DLOAD* used in eca3ghp5.inp.
eca3hhp5.inp	CGAX3HT elements.
eca3hhp5.f	User subroutines *DLOAD* and *DFLUX* used in eca3hhp5.inp.
eca3hfp5.inp	CGAX3T elements.
eca3hfp5.f	User subroutines *DLOAD* and *DFLUX* used in eca3hfp5.inp.
eca4gfp5.inp	CGAX4 elements.
eca4gfp5.f	User subroutine *DLOAD* used in eca4gfp5.inp.
eca4ghp5.inp	CGAX4H elements.
eca4ghp5.f	User subroutine *DLOAD* used in eca4ghp5.inp.
eca4hhp5.inp	CGAX4HT elements.
eca4hhp5.f	User subroutines *DLOAD* and *DFLUX* used in eca4hhp5.inp.
eca4grp5.inp	CGAX4R elements.
eca4grp5.f	User subroutine *DLOAD* used in eca4grp5.inp.
eca4gyp5.inp	CGAX4RH elements.
eca4gyp5.f	User subroutine *DLOAD* used in eca4gyp5.inp.
eca4hfp5.inp	CGAX4T elements.
eca4hfp5.f	User subroutines *DLOAD* and *DFLUX* used in eca4hfp5.inp.
eca6gfp5.inp	CGAX6 elements.
eca6gfp5.f	User subroutine *DLOAD* used in eca6gfp5.inp.
eca6ghp5.inp	CGAX6H elements.
eca6ghp5.f	User subroutine *DLOAD* used in eca6ghp5.inp.
eca6gkp5.inp	CGAX6M elements.
eca6gkp5.f	User subroutine *DLOAD* used in eca6gkp5.inp.
eca6glp5.inp	CGAX6MH elements.
eca6glp5.f	User subroutine *DLOAD* used in eca6glp5.inp.
eca8gfp5.inp	CGAX8 elements.

eca8gfp5.f	User subroutine *DLOAD* used in eca8gfp5.inp.
eca8ghp5.inp	CGAX8H elements.
eca8ghp5.f	User subroutine *DLOAD* used in eca8ghp5.inp.
eca8hhp5.inp	CGAX8HT elements.
eca8hhp5.f	User subroutines *DLOAD* and *DFLUX* used in eca8hhp5.inp.
eca8grp5.inp	CGAX8R elements.
eca8grp5.f	User subroutine *DLOAD* used in eca8grp5.inp.
eca8gyp5.inp	CGAX8RH elements.
eca8gyp5.f	User subroutine *DLOAD* used in eca8gyp5.inp.
eca8hfp5.inp	CGAX8T elements.
eca8hfp5.f	User subroutines *DLOAD* and *DFLUX* used in eca8hfp5.inp.

Patch test for plate bending

Products: Abaqus/Standard

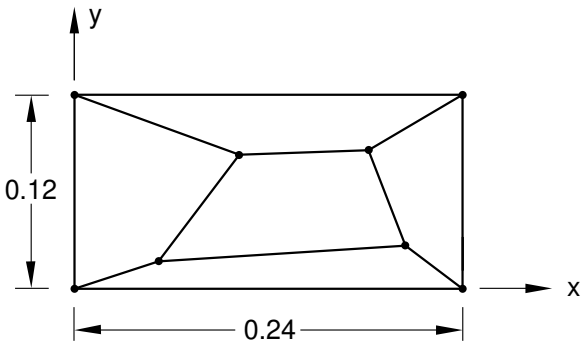
Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

S3 S3R S4 S4R S4R5 S8R S8R5 S9R5 STRI3 STRI65

Problem description



Model: Thickness, $t = 0.001$.

Material: Linear elastic, Young's modulus = 1.0×10^6, Poisson's ratio = 0.25.

Boundary conditions: (applied to all exterior nodes)

$u_x = u_y = 0$, $u_z = 10^{-3}(x^2 + xy + y^2)/2$, $\theta_x = 10^{-3}(y + x/2)$, $\theta_y = -10^{-3}(x + y/2)$.

Reference solution

Stress: $\sigma_x = \sigma_y = \pm 0.6667$, $\sigma_{xy} = \pm 0.20$.

Results and discussion

All elements yield exact solutions except S8R. S8R will pass the patch test if the element shapes are rhombic, but they fail the test for general quadrilaterals.

Input files

esf3sxp3.inp	S3/S3R elements.
ese4sxp3.inp	S4 elements.

esf4sxp3.inp	S4R elements.
es54sxp3.inp	S4R5 elements.
es68sxp3.inp	S8R elements.
es58sxp3.inp	S8R5 elements.
es59sxp3.inp	S9R5 elements.
es63sxp3.inp	STRI3 elements.
es56sxp3.inp	STRI65 elements.

Patch test for beam elements

Products: Abaqus/Standard Abaqus/Explicit

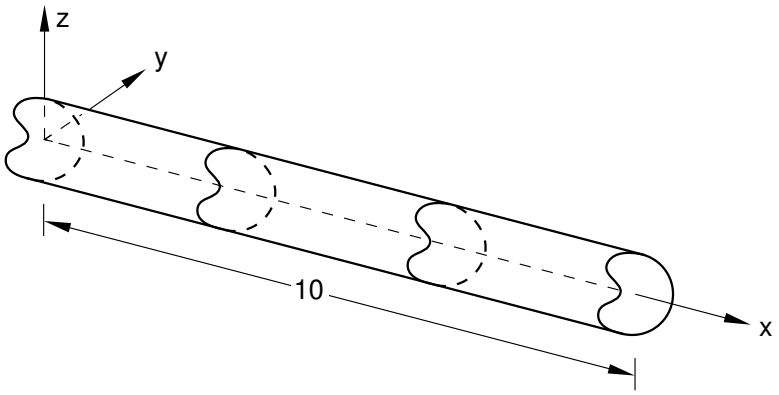
Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

B21 B21H B22 B22H B23 B23H B31 B31H B31OS B31OSH
B32 B32H B32OS B32OSH B33 B33H
PIPE21 PIPE21H PIPE22 PIPE22H PIPE31 PIPE31H PIPE32 PIPE32H

Problem description



Model: Area, $A = 0.01$.

Material: Linear elastic, Young's modulus = 30.0×10^6, Poisson's ratio = 0.3.

Loading and boundary conditions for Step 1: $u_z = \theta_x = \theta_y =$ 0.0 at $x =$ 0, $u_z =$ 0.0 at $x =$ 10; displacement boundary conditions applied at the end nodes: $u_x =$ 0.01 + 0.01x.

Loading and boundary conditions for Step 2: The node at $x =$ 0 is fixed; $u_y =$ 0 at $x =$ 10; concentrated load at the free end: $P_x =$ 3000.

Loading and boundary conditions for Step 3: $u_z = \theta_x = \theta_y =$ 0.0 at $x =$ 0, $u_z =$ 0.0 at $x =$ 10; displacement boundary conditions applied at the end nodes: $u_x =$ 0.01 + 0.01x, where x is the value of the coordinates in the undeformed geometry.

Loading and boundary conditions for Abaqus/Explicit: The node at $x = 0$ is fixed; $u_y = 0$ at $x = 10$; concentrated load at the free end: $P_x = 3000$ using a smooth step amplitude definition. Solution is computed at time 1.0, including geometric nonlinearity.

Reference solution

The analytical results for each step are presented below.

Step 1: PERTURBATION

Section forces: axial force $N_x = 3000$; tip displacement: $u_{tip} = 1.1 \times 10^{-1}$.

Step 2: NLGEOM

Section forces: axial force $N_x = 3000$; tip displacement: $u_{tip} = 1.005 \times 10^{-1}$.

Step 3: PERTURBATION

Section forces: axial force $N_x = 2970$; tip displacement: $u_{tip} = 1.1 \times 10^{-1}$.

Dynamic Step in Abaqus/Explicit:

Section forces: axial force $N_x = 3000$; tip displacement: $u_{tip} = 1.005 \times 10^{-1}$.

Results and discussion

All elements yield exact solutions except the cubic beams, which differ from the analytical solution by about 2% for the NLGEOM step and the subsequent perturbation step. The elements are recommended only for linear analysis. The results for pipe elements in Abaqus/Explicit are the same as those in Abaqus/Standard.

Input files

eb22rxp6.inp	B21 elements.
eb2hrxp6.inp	B21H elements.
eb23rxp6.inp	B22 elements.
eb2irxp6.inp	B22H elements.
eb2arxp6.inp	B23 elements.
eb2jrxp6.inp	B23H elements.
eb32rxp6.inp	B31 elements.
eb3hrxp6.inp	B31H elements.
ebo2ixp6.inp	B31OS elements.
ebohixp6.inp	B31OSH elements.
eb33rxp6.inp	B32 elements.
eb3irxp6.inp	B32H elements.
ebo3ixp6.inp	B32OS elements.
eboiixp6.inp	B32OSH elements.
eb3arxp6.inp	B33 elements.

eb3jrxp6.inp	B33H elements.
ep22pxp6.inp	PIPE21 elements.
ep2hpxp6.inp	PIPE21H elements.
ep23pxp6.inp	PIPE22 elements.
ep2ipxp6.inp	PIPE22H elements.
ep32pxp6.inp	PIPE31 elements.
ep3hpxp6.inp	PIPE31H elements.
ep33pxp6.inp	PIPE32 elements.
ep3ipxp6.inp	PIPE32H elements.
ebmod1p6.inp	External file used to store common portions of the input files for this problem.
ebmod2p6.inp	External file used to store common portions of the input files for this problem.
patch_pipe2d_xpl.inp	PIPE21 elements in Abaqus/Explicit.
patch_pipe3d_xpl.inp	PIPE31 elements in Abaqus/Explicit.

Patch test for heat transfer elements

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

DC1D2 DC1D3
DC2D3 DC2D4 DC2D6 DC2D8
DC3D4 DC3D6 DC3D8 DC3D10 DC3D15 DC3D20
DCAX3 DCAX4 DCAX6 DCAX8
DS3 DS4 DS6 DS8
C3D4T C3D6T C3D8RT C3D8T C3D8IT C3D8IHT C3D10MHT C3D10MT SC8RT
CAX3T CAX4RHT CAX4RT CAX6MT CGAX4RHT CGAX4RT
CPE3T CPE4RHT CPE4RT CPE6MHT CPE6MT
CPEG4RHT CPEG4RT CPEG6MHT CPEG6MT
CPS3T CPS4RT CPS6MT
EC3D8RT

Problem description

The meshes used for the heat transfer tests are the same as those used for the corresponding stress elements, except that the axisymmetric heat transfer elements use a larger radius.

For coupled temperature-displacement elements dummy mechanical properties are prescribed to complete the material definition.

The total simulation time for the Abaqus/Explicit analysis is 20 units. This provides enough time for the transient solution to reach steady-state conditions in this problem.

Boundary conditions: $T = C_0 + C_1 x + C_2 y + C_3 z$, where T is the temperature, C_0 through C_3 are arbitrary constants, and x, y, z denote spatial location. Temperatures are prescribed at every node along the boundary of the mesh. For shell elements z denotes the normal direction to the shell surface.

Reference solution

Fluxes: Since the temperature field is chosen to be linear, it has constant spatial gradients and, thus, has constant fluxes at every integration point.

Results and discussion

All elements yield exact solutions.

Input files

Abaqus/Standard input files

ec12dfp4.inp	DC1D2 elements.

ec13dfp4.inp	DC1D3 elements.
ec23dfp4.inp	DC2D3 elements.
ec24dfp4.inp	DC2D4 elements.
ec26dfp4.inp	DC2D6 elements.
ec28dfp4.inp	DC2D8 elements.
ec34dfp4.inp	DC3D4 elements.
ec36dfp4.inp	DC3D6 elements.
ec38dfp4.inp	DC3D8 elements.
ec3adfp4.inp	DC3D10 elements.
ec3fdfp4.inp	DC3D15 elements.
ec3kdfp4.inp	DC3D20 elements.
eca3dfp4.inp	DCAX3 elements.
eca4dfp4.inp	DCAX4 elements.
eca6dfp4.inp	DCAX6 elements.
eca8dfp4.inp	DCAX8 elements.
es33dxp4.inp	DS3 elements.
es34dxp4.inp	DS4 elements.
es36dxp4.inp	DS6 elements.
es38dxp4.inp	DS8 elements.
ec38trp4.inp	C3D8RT elements.
heatpatch_std_c3d8it.inp	C3D8IT elements.
heatpatch_std_c3d8iht.inp	C3D8IHT elements.
ec3atlp4.inp	C3D10MHT elements.
ec3atkp4.inp	C3D10MT elements.
eca4typ4.inp	CAX4RHT elements.
eca4trp4.inp	CAX4RT elements.
eca4hyp4.inp	CGAX4RHT elements.
eca4hrp4.inp	CGAX4RT elements.
ece4trp4.inp	CPE4RT elements.
ece6tlp4.inp	CPE6MHT elements.
ece6tkp4.inp	CPE6MT elements.
ecg4typ4.inp	CPEG4RHT elements.
ecg4trp4.inp	CPEG4RT elements.
ecg6tlp4.inp	CPEG6MHT elements.
ecg6tkp4.inp	CPEG6MT elements.
ecs4trp4.inp	CPS4RT elements.
ecs6tkp4.inp	CPS6MT elements.

Abaqus/Explicit input files

heatpatch_xpl_c3d4t.inp	C3D4T elements.
heatpatch_xpl_c3d6t.inp	C3D6T elements.
heatpatch_xpl_c3d8rt.inp	C3D8RT elements.
heatpatch_xpl_sc8rt.inp	SC8RT elements.

heatpatch_xpl_c3d8t.inp	C3D8T elements.
heatpatch_xpl_c3d10mt.inp	C3D10MT elements.
heatpatch_xpl_cax3t.inp	CAX3T elements.
heatpatch_xpl_cax4rt.inp	CAX4RT elements.
heatpatch_xpl_cax6mt.inp	CAX6MT elements.
heatpatch_xpl_cpe3t.inp	CPE3T elements.
heatpatch_xpl_cpe4rt.inp	CPE4RT elements.
heatpatch_xpl_cpe6mt.inp	CPE6MT elements.
heatpatch_xpl_cps3t.inp	CPS3T elements.
heatpatch_xpl_cps4rt.inp	CPS4RT elements.
heatpatch_xpl_cps6mt.inp	CPS6MT elements.
htpatch_xpl_ec3d8rt.inp	EC3D8RT elements.

Patch test for thermal-electrical elements

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

DCAX3E DCAX4E DCAX6E DCAX8E
DC2D3E DC2D4E DC2D6E DC2D8E
DC3D4E DC3D6E DC3D8E DC3D10E DC3D15E DC3D20E

Problem description

The meshes used for the thermal-electrical element tests are the same as those used for the corresponding heat transfer elements.

Boundary conditions: $T = C_0 + C_1 x + C_2 y + C_3 z$, where T is the temperature, C_0 through C_3 are arbitrary constants, and x, y, z denote spatial location. $\phi = D_0 + D_1 x + D_2 y + D_3 z$, where ϕ is the electrical potential, D_0 through D_3 are arbitrary constants, and x, y, z denote spatial location. Temperature and electrical potentials are prescribed at every node along the boundary of the mesh.

Reference solution

Fluxes: Since the temperature and electrical potential fields are chosen to be linear, they have constant spatial gradients and, thus, have constant fluxes at every integration point.

Results and discussion

All elements yield exact solutions.

Input files

eca3vfpj.inp	DCAX3E elements.
eca4vfpj.inp	DCAX4E elements.
eca6vfpj.inp	DCAX6E elements.
eca8vfpj.inp	DCAX8E elements.
ec23vfpj.inp	DC2D3E elements.
ec24vfpj.inp	DC2D4E elements.
ec26vfpj.inp	DC2D6E elements.
ec28vfpj.inp	DC2D8E elements.
ec34vfpj.inp	DC3D4E elements.
ec36vfpj.inp	DC3D6E elements.
ec38vfpj.inp	DC3D8E elements.
ec3avfpj.inp	DC3D10E elements.

ec3fvfpj.inp	DC3D15E elements.
ec3kvfpj.inp	DC3D20E elements.

Patch test for acoustic elements

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

AC1D2 AC1D3
ACAX3 ACAX4 ACAX4R ACAX6 ACAX8
AC2D3 AC2D4 AC2D4R AC2D6 AC2D8
AC3D4 AC3D5 AC3D6 AC3D8 AC3D8R AC3D10 AC3D15 AC3D20

Problem description

The meshes used for the acoustic element patch tests are the same as those used for the corresponding heat transfer elements.



Note:

The models are analyzed via steady-state dynamic procedures in which a small frequency, 0.01 Hz, is requested. In Abaqus/Explicit the steady-state results are obtained by performing a long-term transient simulation.

Boundary conditions: $P = C_0 + C_1 x + C_2 y + C_3 z$, where P is the acoustic pressure, C_0 through C_3 are arbitrary constants, and x, y, z denote spatial location. Acoustic pressures (DOF 8) are prescribed at every node along the boundary of the mesh.

Reference solution

It is currently not possible to report the pressure gradients for acoustic elements in Abaqus. However, it is possible to compare the acoustic pressures at the interior nodes of the mesh to the values that are analytically calculated from the above expression for P.

Results and discussion

All elements yield exact values of P at the interior nodes of the models.

Input files

Abaqus/Standard input files

ec12afp7.inp	AC1D2 elements.
ec13afp7.inp	AC1D3 elements.
eca3afp7.inp	ACAX3 elements.
eca4afp7.inp	ACAX4 elements.

eca6afp7.inp	ACAX6 elements.
eca8afp7.inp	ACAX8 elements.
ec23afp7.inp	AC2D3 elements.
ec24afp7.inp	AC2D4 elements.
ec26afp7.inp	AC2D6 elements.
ec28afp7.inp	AC2D8 elements.
ec34afp7.inp	AC3D4 elements.
ec35afp7.inp	AC3D5 elements.
ec36afp7.inp	AC3D6 elements.
ec38afp7.inp	AC3D8 elements.
ec3aafp7.inp	AC3D10 elements.
ec3fafp7.inp	AC3D15 elements.
ec3kafp7.inp	AC3D20 elements.

Abaqus/Explicit input files

acousticpatch_xpl_acax3.inp	ACAX3 elements.
acousticpatch_xpl_acax4r.inp	ACAX4R elements.
acousticpatch_xpl_ac2d3.inp	AC2D3 elements.
acousticpatch_xpl_ac2d4r.inp	AC2D4R elements.
acousticpatch_xpl_ac3d4.inp	AC3D4 elements.
acousticpatch_xpl_ac3d6.inp	AC3D6 elements.
acousticpatch_xpl_ac3d8r.inp	AC3D8R elements.

Connector element verification

In this section:

- *Damped free vibration with initial conditions*
- *Sinusoidal excitation of a damped spring-mass system*
- *Multiple instances of connector elements*
- *Individual connector option tests*
- *Connector elements in perturbation analyses*
- *Tests for special-purpose connectors*

Damped free vibration with initial conditions

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

CONN2D2 CONN3D2

Problem description

Translational and rotational basic connector components as well as translational-rotational combination connections are tested in damped, free vibration analyses. A series of simple, 2-node connector element models verify the connections described in *Table 1*. Two-dimensional, three-dimensional, and axisymmetric cases are examined where appropriate. In each case all available components of relative motion produce the same free, damped sinusoidal response based on the initial displacements. In Abaqus/Standard linear geometry is used to avoid nonlinear coupling for finite rotation connection types.

The verification analyses consist of two steps in Abaqus/Standard and one step in Abaqus/Explicit. The initial displacements for the dynamic response are defined as follows. The first step in the Abaqus/Standard analysis prescribes the connector motion with a static step to apply an initial displacement to the “free” end node of the connector. The Abaqus/Explicit analysis defines reference lengths and angles for constitutive response that are different from those calculated from the initial geometry. In the dynamic analysis the free node end of the connector element undergoes damped, free vibration in response to the initial displacement.

Table 1: Connection types for damped, free vibration verification analyses.

Connection Type	Degrees of Freedom Affected	Dimensions
AXIAL	Translational Basic	2D, 3D, Axisymmetric
CARTESIAN	Translational Basic	2D, 3D, Axisymmetric
PROJECTION CARTESIAN	Translational Basic	3D
RADIAL-THRUST	Translational Basic	3D
SLIDE-PLANE	Translational Basic	3D
SLOT	Translational Basic	2D, 3D, Axisymmetric
CARDAN	Rotational Basic	3D
EULER	Rotational Basic	3D
FLEXION-TORSION	Rotational Basic	3D
PROJECTION FLEXION-TORSION	Rotational Basic	3D
REVOLUTE	Rotational Basic	3D
ROTATION	Rotational Basic	2D, 3D, Axisymmetric
UNIVERSAL	Rotational Basic	3D
AXIAL/EULER	Combination	3D
CARTESIAN/CARDAN	Combination	3D

Connection Type	Degrees of Freedom Affected	Dimensions
CARTESIAN/ROTATION	Combination	2D, 3D, Axisymmetric
JOIN/FLEXION-TORSION	Combination	3D
LINK/UNIVERSAL	Combination	3D
RADIAL-THRUST/ CONSTANT VELOCITY	Combination	3D
SLIDE-PLANE/REVOLUTE	Combination	3D
SLOT/ALIGN	Combination	2D, 3D, Axisymmetric
PROJECTION CARTESIAN/PROJECTION FLEXION-TORSION	Combination	3D

Model:

Initial relative displacement	0.1
Initial relative rotation	0.1

Material:

Translational spring stiffness	48.0
Translational damping coefficient	5.0
Mass	12.0
Torsional spring stiffness	480.0
Torsional damping coefficient	50.0
Rotary inertia (isotropic)	120.0

Results and discussion

Abaqus provides the expected solution for all cases.

Input files

Abaqus/Standard input files

conn_std_free_2d.inp	Connector elements in two dimensions.
conn_std_free_3d.inp	Connector elements in three dimensions.
conn_std_free_axi.inp	Connector elements in axisymmetric analyses.
conn_std_free_bushing.inp	Bushing element.

Abaqus/Explicit input files

conn_xpl_free_2d.inp	Connector elements in two dimensions.
conn_xpl_free_3d.inp	Connector elements in three dimensions.
conn_xpl_free_axi.inp	Connector elements in axisymmetric analyses.

Sinusoidal excitation of a damped spring-mass system

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

CONN2D2 CONN3D2

Problem description

Translational and rotational basic connector components as well as translational-rotational combination connections are tested for sinusoidal excitation of a damped spring-mass connector system. A series of simple, 2-node connector elements simulating a damped spring-mass system verify the connections described in *Table 1*. The kinetic behavior in the connector elements is modeled using the connector elasticity, connector damping, and connector friction behaviors.

In Abaqus/Standard a direct-integration implicit dynamic step is used with a small- or large-displacement formulation. In Abaqus/Explicit the usual explicit dynamic procedure is used with the default large-displacement formulation. The spring-mass system in each case is driven by assigning a connector element load.

Table 1: Connection types for sinusoidal excitation verification analyses.

Connection Type	Degrees of Freedom Affected	Dimensions
AXIAL	Translational Basic	2D, 3D, Axisymmetric
CARTESIAN	Translational Basic	2D, 3D, Axisymmetric
PROJECTION CARTESIAN	Translational Basic	3D
RADIAL-THRUST	Translational Basic	3D
SLIDE-PLANE	Translational Basic	3D
SLOT	Translational Basic	2D, 3D, Axisymmetric
CARDAN	Rotational Basic	3D
EULER	Rotational Basic	3D
FLEXION-TORSION	Rotational Basic	3D
PROJECTION FLEXION-TORSION	Rotational Basic	3D
REVOLUTE	Rotational Basic	3D
ROTATION	Rotational Basic	2D, 3D, Axisymmetric
UNIVERSAL	Rotational Basic	3D
AXIAL/EULER	Combination	3D
CARTESIAN/CARDAN	Combination	3D
CARTESIAN/ ROTATION	Combination	3D
JOIN/ FLEXION-TORSION	Combination	3D
LINK/UNIVERSAL	Combination	3D

Connection Type	Degrees of Freedom Affected	Dimensions
RADIAL-THRUST/ CONSTANT VELOCITY	Combination	3D
SLIDE-PLANE/REVOLUTE	Combination	3D
SLOT/ALIGN	Combination	3D
PROJECTION CARTESIAN/PROJECTION FLEXION-TORSION	Combination	3D

Model:

Connector load for relative translations	4.8
Connector load for relative rotations	48.0

Material:

Translational spring stiffness	48.0
Translational damping coefficient	5.0
Mass	12.0
Torsional spring stiffness	480.0
Torsional damping coefficient	50.0
Rotary inertia (isotropic)	120.0

Results and discussion

Abaqus provides the expected solution.

Input files

Abaqus/Standard input files

conn_std_force_2d.inp	Connection in two dimensions.
conn_std_force_2d_fric.inp	Connection in two dimensions with friction.
conn_std_force_3d.inp	Connection in three dimensions.
conn_std_force_3d_fric.inp	Connection in three dimensions with friction.
conn_std_force_axi.inp	Connection in an axisymmetric model.
conn_std_force_axi_fric.inp	Connection in an axisymmetric model with friction.
conn_std_force_bushing.inp	Bushing element.

Abaqus/Explicit input files

conn_xpl_force_2d.inp	Connection in two dimensions.
conn_xpl_force_2d_fric.inp	Connection in two dimensions with friction.
conn_xpl_force_3d.inp	Connection in three dimensions.
conn_xpl_force_3d_fric.inp	Connection in three dimensions with friction.
conn_xpl_force_axi.inp	Connection in an axisymmetric model.
conn_xpl_force_axi_fric.inp	Connection in an axisymmetric model with friction.

Multiple instances of connector elements

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Actuation of connected rigid bodies

Elements tested

CONN3D2

Problem description

This verification problem tests the option of specifying connector motion to prescribe the relative motions of an articulated structure. A robotic-like crane assembly, modeled as rigid bodies connected together by means of connector elements, is subjected to actuating motions that drive the kinematic connections by specified amplitude curves. The actuating motions, including relative sliding and a two-axes rotation, cause the assembly to open up in a smooth sequence to form a riser crane. After a drilling and downward motion of the outermost body, the assembly closes down and reverts to its starting configuration. Tests are conducted both with no friction and with frictional effects in the connections.

Model: The model consists of rigid bodies and connector elements as described in the table below. Each rigid body pair in the table is connected by rotational and translational basic connector types with connector motion definitions in each of the available relative components of motion.

Table 1: Rigid bodies and connectors.

Body 1	Body 2	Basic Connector Types	
		Translational	Rotational
Base	Arm 1	SLOT	REVOLUTE
Arm 1	Cover	JOIN	REVOLUTE
Arm 1	Arm 2	SLOT	ALIGN
Arm 2	Riser 1	CARTESIAN	CARDAN
Riser 1	Riser 2	SLOT	ALIGN
Riser 2	Crane	JOIN	REVOLUTE
Crane	Chuck	JOIN	REVOLUTE
Chuck	Bit	CARTESIAN	CARDAN

The complete model in the fully open configuration with the rigid bodies labeled is shown in *Figure 1*.

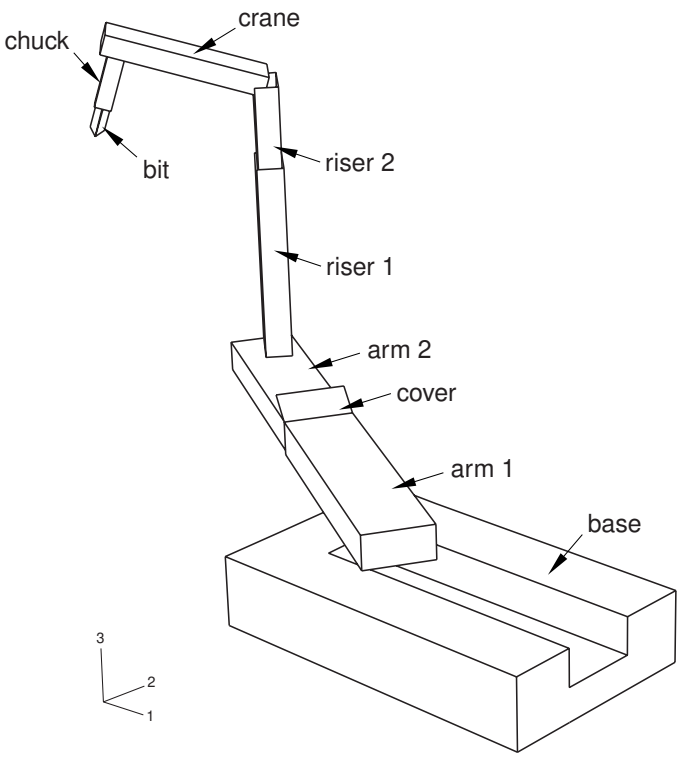


Figure 1: Crane assembly in fully open configuration.

Results and discussion

Abaqus provides the expected solution for all cases.

Input files

conn_std_craneactuation.inp	Abaqus/Standard input file.
conn_std_craneactuation_fric.inp	Abaqus/Standard input file with friction.
conn_xpl_craneactuation.inp	Abaqus/Explicit input file.
conn_xpl_craneactuation_fric.inp	Abaqus/Explicit input file with friction.

Marbles in a jar

Problem description

This problem, which is analyzed using only Abaqus/Explicit, tests a connector stop for multiple intermittent contacts. Eight rigid spheres (marbles) are dropped into a rigid container (jar). The marbles move down through the jar and, after some jostling, come to rest in an equilibrium position at the bottom of the jar. The interaction between the marbles is modeled by defining a connector element for each marble pair, while the interaction between the marbles and the jar is modeled by defining a connector element between each marble and the jar.

Model: The jar and marbles are each modeled as rigid bodies. An analytical rigid surface of revolution is defined for each marble to represent the spherical outer surface for visualization purposes only. Each marble is dropped into the jar by defining an initial velocity in the direction of the axis of the jar and specifying a force

on each rigid body reference node to simulate gravity. AXIAL connector types are defined for each pair of marbles, with the connector stop used to constrain the motion of each pair so that the marbles in the pair do not overlap. RADIAL-THRUST connector types are defined between each marble and the jar. These connectors constrain the motion of each marble so that the marble remains in the interior of the jar (i.e., it does not slip through the side walls or fall through the bottom of the jar) by using the connector stop.

The marbles and jar in their initial and final configurations are shown in *Figure 2* and *Figure 3*.

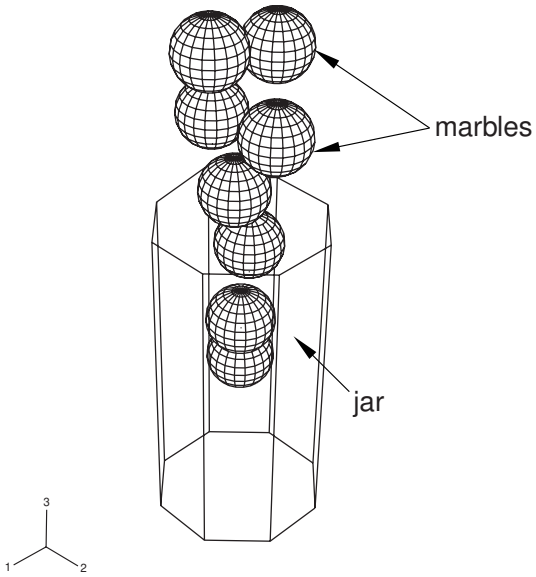


Figure 2: Marbles and jar in initial configuration.

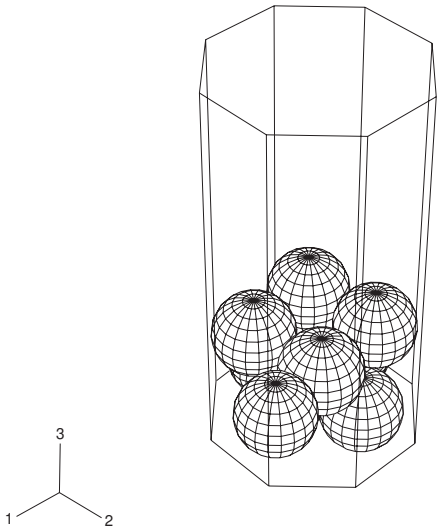


Figure 3: Marbles and jar in final configuration.

Results and discussion

Abaqus/Explicit provides the expected solution for all cases.

Input files

conn_xpl_marblesinjar.inp Abaqus/Explicit input file.

Satellite deployment

Problem description

This problem, which is analyzed using both Abaqus/Explicit and Abaqus/Standard, tests the connector lock procedure for an articulated deformable structure. The complex sequence of motions analyzed is similar to that of a spinning satellite, with flexible boom arms, during its deployment. The satellite here consists of a core with large mass and rotary inertia and three comparatively light articulated arms. The arms undergo a series of large translations and rotations before reaching their final deployment position when they are locked into place. The connections between the components of each arm and between the arms and the satellite core are modeled with connector elements.

Model: The satellite core is modeled as a rigid body. The booms consist of three parts—the inner arm, the middle arm, and the outer arm—and are modeled with elastic beam elements. The satellite core is connected to each inner arm by means of a JOIN and a REVOLUTE connection. Each inner arm in turn is connected to its corresponding middle arm using the same translational and rotational connection types. Each middle arm is connected similarly to its corresponding outer arm. An initial rotating velocity about the global z-axis is specified for the entire model. In each of the connections described above, the rotations about the local 1-axis are constrained to lock into place once they reach their final deployment value of 180° using a connector lock. In addition, torsional springs are defined in the connections between the inner arms and middle arms and between the middle arms and outer arms using connector elasticity. The torsional springs act in addition to the centrifugal force to help the arms reach their final deployed configuration. Tests are conducted both with no friction and with frictional effects in the connections.

The complete model in the initial and final configurations is shown in *Figure 4* and *Figure 5*.

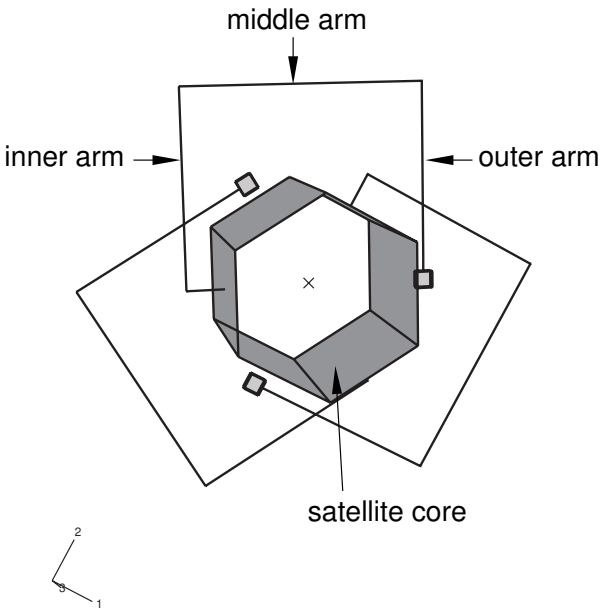


Figure 4: Satellite in initial configuration.

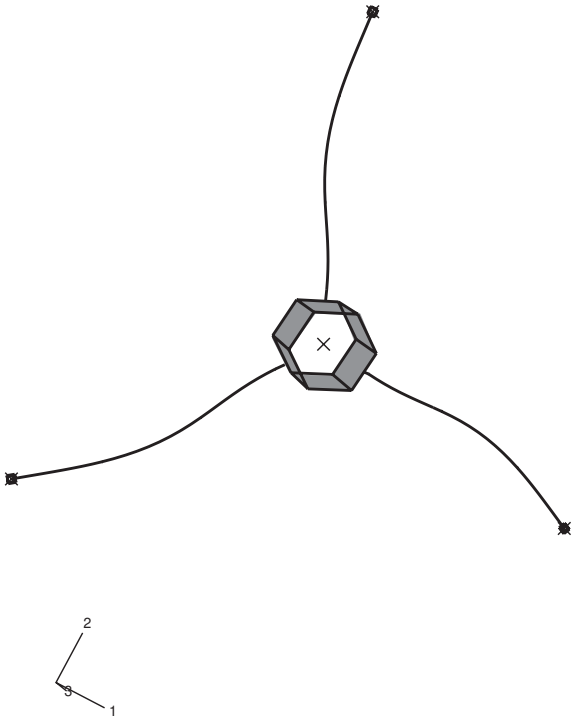


Figure 5: Satellite in final deployed configuration.

Results and discussion

Abaqus provides the expected solution for all cases.

Input files

conn_std_satellitedeploy.inp	Abaqus/Standard input file.
conn_std_satellitedeploy_fric.inp	Abaqus/Standard input file with friction.
conn_xpl_satellitedeploy.inp	Abaqus/Explicit input file.
conn_xpl_satellitedeploy_fric.inp	Abaqus/Explicit input file with friction.

Abacus subjected to prescribed motion

Problem description

This problem, which is analyzed using only Abaqus/Explicit, tests the connector stop procedure for multiple intermittent contacts and kinematic constraints. An abacus consisting of a frame and beads is modeled. As the frame undergoes large motions, the beads slide up and down the sliders in the frame. Connector elements are used to model the contact interactions between the beads, the contact interactions between the beads and the frame, and the kinematic constraints between the beads and the frame.

Model: The abacus frame, consisting of sliders and a separator, is modeled as a single rigid body. Each of the beads is modeled as a rigid body, and an analytical rigid surface of revolution is used to model the surface of the bead for visualization purposes only. The frame is subjected to prescribed translations and rotations by means of specified amplitude curves. AXIAL connector types are defined between adjacent beads on the same slider, with the connector stop used to constrain the relative sliding motion between adjacent beads so that the beads

do not overlap. Each bead is also connected to the frame by defining connector elements using the SLOT and ALIGN basic connection types. These elements ensure that each bead moves along its slider and rotates with the frame. The connector stop is specified for the connector elements between the frame and the beads next to the separator. The connector stop is also specified for the connector elements between the frame and the beads at the extreme end of each slider. These connector stops ensure that the beads slide only along the length of their respective sliders and prevent the beads from leaving the sliders.

The abacus in its initial, final, and two intermediate configurations is shown in *Figure 6*, *Figure 7*, *Figure 8*, and *Figure 9*.

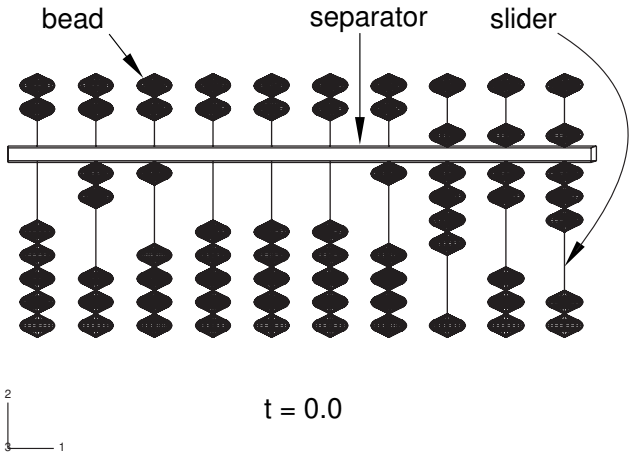


Figure 6: Abacus at time t = 0.0.

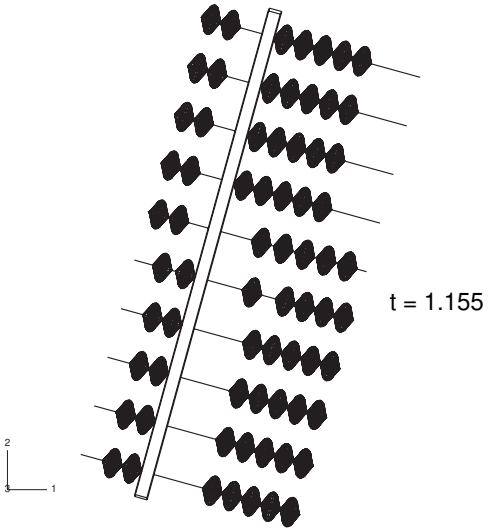


Figure 7: Abacus at time t = 1.155.

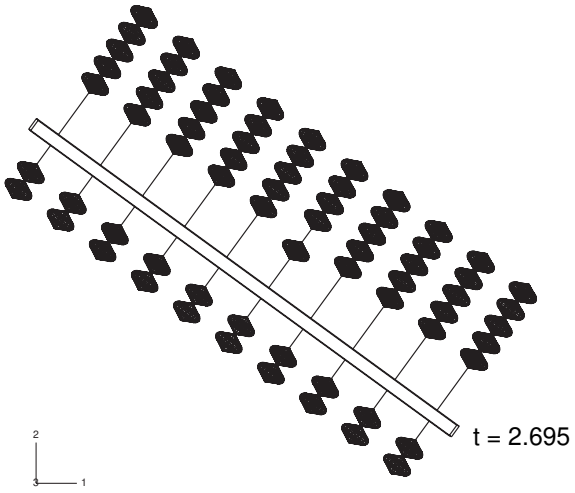


Figure 8: Abacus at time t = 2.695.

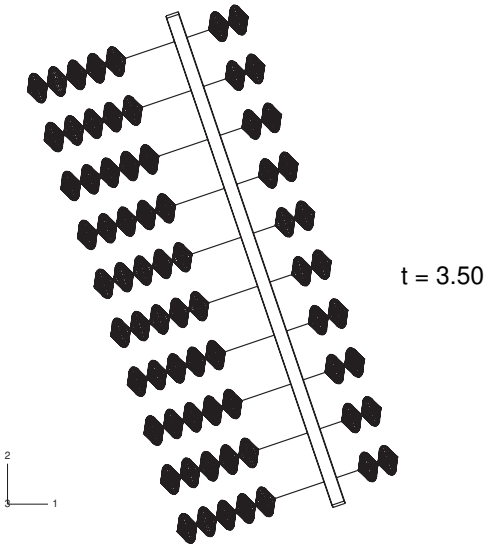


Figure 9: Abacus at time t = 3.50.

Results and discussion

Abaqus/Explicit provides the expected solution in all cases.

Input files

conn_xpl_abacusmotion.inp	Abaqus/Explicit input file.

Individual connector option tests

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elastic and damped connector behavior

Elements tested

CONN2D2 CONN3D2

Problem description

These verification cases test the performance of connector behavior options not routinely used in other verification problems. This section focuses on spring or damper behaviors via the connector elasticity and connector damping options. Both CARTESIAN and CARDAN connections are employed in these verification cases.

The behavior options are verified by applying a concentrated load with a connector element load and achieving a resulting relative displacement (for connector elasticity) or velocity (for connector damping) that corresponds to an analytical solution. Equivalent, nonconnector elements are included for comparison.

For both the CARTESIAN and CARDAN connections the following connector elasticity cases are tested:

1. Connector elasticity with the connector's component of relative motion for which elastic behavior is specified set to 1 (CARTESIAN) or 4 (CARDAN) and the number of field variable dependencies set to 2 with the following dependency settings:
   a. Temperature = −10, field variable 1 = 1.0, field variable 2 = 0.5
   b. Temperature = 90, field variable 1 = 2.0, field variable 2 = 1.0
2. Connector elasticity with the connector's component of relative motion for which elastic behavior is specified set to 1 (CARTESIAN) or 4 (CARDAN), the number of field variable dependencies set to 1, and nonlinear behavior defined with the following dependency settings:
   a. Temperature = −10, field variable 1 = 1.0
   b. Temperature = 90, field variable 1 = 2.0
3. Connector elasticity with the connector's component of relative motion for which elastic behavior is specified set to 1 (CARTESIAN) or 4 (CARDAN), dependencies on components of constitutive motion included in the elasticity definition, and nonlinear behavior defined (no relevant temperature or field variable dependencies)
4. Connector elasticity with the connector's component of relative motion for which elastic behavior is specified set to 1 (CARTESIAN) or 4 (CARDAN), dependencies on components of relative position included in the elasticity definition, and nonlinear behavior and a periodic model defined (no relevant temperature or field variable dependencies)
5. Connector elasticity with the number of field variable dependencies included in the definition of connector elasticity data set to 2 (coupled)
   a. Field variable 1 = 1.0, field variable 2 = 0.5
   b. Field variable 1 = 2.0, field variable 2 =1.0

Similarly, for both the CARTESIAN and CARDAN connections the following connector damping cases are tested:

1. Connector damping with the connector's component of relative motion for which damping behavior is specified set to 1 (CARTESIAN) or 4 (CARDAN) and the number of field variable dependencies set to 2 with the following dependency settings:
   a. Temperature = −10, field variable 1 = 1.0, field variable 2 = 0.5
   b. Temperature = 90, field variable 1 = 2.0, field variable 2 = 1.0
2. Connector damping with the connector's component of relative motion for which damping behavior is specified set to 1 (CARTESIAN) or 4 (CARDAN), the number of field variable dependencies set to 1, and nonlinear behavior defined with the following dependency settings:
   a. Temperature = −10, field variable 1 = 1.0
   b. Temperature = 90, field variable 1 = 2.0
3. Connector damping with the connector's component of relative motion for which damping behavior is specified set to 1 (CARTESIAN), dependencies on components of relative position included in the damping definition, and nonlinear behavior and a periodic model defined (no relevant temperature or field variable dependencies)
4. Connector damping with the number of field variable dependencies included in the definition at connector damping data set to 1 (coupled) with the following dependency settings:
   a. Field variable 1 = 1.0
   b. Field variable 1 = 2.0

Model: The models consist of a series of independent, 2-node connector elements with relevant connector behaviors.

Results and discussion

Abaqus matches the analytical solution for all relevant dependency settings.

Input files

Abaqus/Standard input files

misc_elascart_std_conn2d.inp	Elastic connector behavior.
misc_elascart_std_conn3d.inp	Elastic connector behavior.
misc_elascardan_std_conn3d.inp	Elastic connector behavior.
misc_dampcart_std_conn2d.inp	Damping connector behavior.
misc_dampcart_std_conn3d.inp	Damping connector behavior.
misc_dampcardan_std_conn3d.inp	Damping connector behavior.

Abaqus/Explicit input files

misc_elascart_xpl_conn2d.inp	Elastic connector behavior.
misc_elascart_xpl_conn3d.inp	Elastic connector behavior.
misc_elascardan_xpl_conn3d.inp	Elastic connector behavior.
misc_dampcart_xpl_conn2d.inp	Damping connector behavior.
misc_dampcart_xpl_conn3d.inp	Damping connector behavior.

misc_dampcart_xpl_conn3d_irreg.inp	Damping connector behavior, no regularization of tabular data.
misc_dampcardan_xpl_conn3d.inp	Damping connector behavior.

Connector rigid behavior

Elements tested

CONN3D2

Problem description

These verification cases test the connector rigid behavior. Both CARTESIAN and CARDAN connections are used.

The behavior options are verified by applying a concentrated load via a nodal concentrated load option, such that some force is created in the connector. Equivalent models with intrinsically constrained components of relative motion are created, and the results are compared.

Results and discussion

The results from analyses using the connector rigid behavior match the results from analyses using intrinsically constrained components.

Input files

Abaqus/Standard input files

misc_rigcart_std_conn3d.inp	Rigid connector behavior.
misc_rigcard_std_conn3d.inp	Rigid connector behavior.

Abaqus/Explicit input files

misc_rigcart_xpl_conn3d.inp	Rigid connector behavior.
misc_rigcard_xpl_conn3d.inp	Rigid connector behavior.

Connector plastic behavior

Elements tested

CONN2D2 CONN3D2

Problem description

These verification cases test the connector elastic-plastic and rigid-plastic behavior defined using the connector plasticity and hardening behaviors (both with and without mode mix ratio dependency) in association with procedures specifying connector elasticity and user-defined potentials in connector elements. An assembled connection using the basic connection types CARTESIAN and CARDAN is used for most cases. The assembled connection type BUSHING is used for mode mix ratio–dependent connector hardening behavior. For the two-dimensional analyses, a CARTESIAN connection is used.

The behavior options are verified by applying a concentrated load with a connector element load and achieving a resulting relative motion or relative plastic motion that corresponds to an analytical solution.

Results and discussion

Abaqus matches the analytical solution for all relevant settings.

Input files

Abaqus/Standard input files

misc_linelasplas_std_conn3d.inp	Linear elastic-plastic connector behavior.
misc_linelasplas_std_conn2d.inp	Linear elastic-plastic connector behavior.
misc_rigplas_std_conn3d.inp	Rigid-plastic connector behavior.
misc_nonlinelasplas_std_conn3d.inp	Nonlinear elastic-plastic connector behavior.
misc_conn_plasdamage_spotweld_ modemix-1.inp	Plastic connector behavior with damage using BUSHING connection, mode mix ratio–dependent connector hardening.
misc_conn_plasdamage_spotweld_ modemix-2.inp	Plastic connector behavior with damage using BUSHING connection, mode mix ratio–dependent connector hardening.

Abaqus/Explicit input files

misc_linelasplas_xpl_conn3d.inp	Linear elastic-plastic connector behavior.
misc_linelasplas_xpl_conn3d_irreg.inp	Linear elastic-plastic connector behavior, no regularization of tabular data.
misc_linelasplas_xpl_conn2d.inp	Linear elastic-plastic connector behavior.
misc_rigplas_xpl_conn3d.inp	Rigid-plastic connector behavior.
misc_nonlinelasplas_xpl_conn3d.inp	Nonlinear elastic-plastic connector behavior.
misc_conn_plasdamage_spotweld_modemix_ xpl1.inp	Plastic connector behavior with damage using the BUSHING connection, mode mix ratio–dependent connector hardening.
misc_conn_plasdamage_spotweld_modemix_ xpl2.inp	Plastic connector behavior with damage using the BUSHING connection, mode mix ratio–dependent connector hardening.

Connector damage behavior

Elements tested

CONN2D2 CONN3D2

Problem description

These verification cases test the connector elastic (linear and nonlinear) and rigid-plastic behavior with damage defined using connector damage initiation and evolution in association with the procedures specifying connector elasticity, connector plasticity, and connector hardening behaviors. An assembled connection using the basic connection types CARTESIAN and CARDAN is used for all cases except one case where the assembled connection type BUSHING is used. For the two-dimensional analyses, a CARTESIAN connection is used.

Results and discussion

Abaqus matches the expected solutions for all relevant settings.

Input files

Abaqus/Standard input files

misc_nonlinelas_dam_std_conn3d.inp	Nonlinear elastic with damage connector behavior.
misc_nonlinelas_dam_std_conn2d.inp	Nonlinear elastic with damage connector behavior.
misc_rigplas_dam_std_conn3d.inp	Rigid-plastic with damage connector behavior.
misc_rigplas_dam_std_bushing.inp	Rigid-plastic with damage connector behavior using the BUSHING connection.
misc_rigplas_dam_std_bushing_xtrpl.inp	Rigid-plastic with damage connector behavior using the BUSHING connection, linear extrapolation.
misc_conn_plasdamage_spotweld.inp	Plastic with damage connector behavior using the BUSHING connection.
misc_elas_dam_mult_std_conn3d.inp	Linear elastic with multiple damage mechanisms connector behavior.
misc_dam_sectcontrol_std.inp	Section control options for connector damage.

Abaqus/Explicit input files

misc_nonlinelas_dam_xpl_conn3d.inp	Nonlinear elastic with damage connector behavior.
misc_nonlinelas_dam_xpl_conn2d.inp	Nonlinear elastic with damage connector behavior.
misc_nonlinelas_dam_xpl_conn2d_irreg.inp	Nonlinear elastic with damage connector behavior, no regularization of tabular data.
misc_rigplas_dam_xpl_conn3d.inp	Rigid-plastic with damage connector behavior.
misc_rigplas_dam_xpl_bushing.inp	Rigid-plastic with damage connector behavior using the BUSHING connection.
misc_rigplas_dam_xpl_bushing_xtrpl.inp	Rigid-plastic with damage connector behavior using the BUSHING connection, linear extrapolation.
misc_rigplas_dam_xpl_irregxtrpl.inp	Rigid-plastic with damage connector behavior using the BUSHING connection, no regularization of tabular data, linear extrapolation.
misc_elas_dam_mult_xpl_conn3d.inp	Linear elastic with multiple damage mechanisms connector behavior.

Connector uniaxial behavior

Elements tested

CONN2D2 CONN3D2

Problem description

These verification cases test the connector uniaxial behavior defined by prescribing the loading/unloading response for the component of relative motion using uniaxial behavior loading and unloading data. An AXIAL connection type is employed in these verification cases.

The behavior options are verified by applying a concentrated load and achieving a resulting relative motion that corresponds to the prescribed loading/unloading response.

Results and discussion

Abaqus matches the prescribed loading/unloading response for all relevant settings.

Input files

Abaqus/Explicit input files

misc_uniaxial_damage_xpl_conn3d.inp	Connector uniaxial behavior models with damage.
misc_uniaxial_permdeform_xpl_conn3d.inp	Connector uniaxial behavior models with permanent deformation.
misc_uniaxial_ratedep.inp	Rate-dependent connector uniaxial behavior.
misc_uniaxial_mixed_xpl_conn3d.inp	Combining different uniaxial behavior models in tension and compression.
misc_uniaxial_indepcomp_xpl_conn3d.inp	Connector uniaxial behavior where the loading/unloading response depends on relative motion in several component directions.
misc_uniaxial_mixed_onset_xpl_conn3d.inp	Combining different uniaxial behavior models in tension and compression; tensile behavior exhibiting permanent deformation after a specified onset of yield and compressive behavior with damage after a specified onset of damage.

Conditional position-dependent connector behavior

Elements tested

CONN2D2 CONN3D2

Problem description

This section focuses on stopping and locking behaviors defined with connector stop and connector lock definitions. Both CARTESIAN and CARDAN connections are used.

The behavior options are verified through a two-step load history. In Step 1 a concentrated load is applied with the connector load, such that the resulting connector motion will exceed the prescribed motion limits for either the connector stop or lock. In Step 2 the load direction is reversed to confirm the stopping or locking behavior. Equivalent, nonconnector elements are included for comparison. In the Abaqus/Standard tests a linear perturbation static step is performed in the third step.

For CARTESIAN connections the following connector lock cases are tested:

1. Connector lock with the component number on which the locking criterion is based set to 1 and all components of relative motion locked when the locking criterion is satisfied.
2. Connector lock with the component number on which the locking criterion is based set to 3 and the component number to lock, when the locking criterion is satisfied, set to 2.

For CARDAN connections, the following connector lock cases are tested:

1. Connector lock with the component number on which the locking criterion is based set to 4 and the component number to lock, when the locking criterion is satisfied, set to 4 (with rotation criterion) or 6 (with moment criterion)
2. Connector lock with the component number on which the locking criterion is based set to 4 and the component number to lock, when the locking criterion is satisfied, set to 4 (with rotation criterion) or 6 (with moment criterion)

For CARTESIAN and CARDAN connections, the following connector stop cases are tested:

1. Connector stop with the component number, for which the connector stop is defined, set to 1 (CARTESIAN) or 4 (CARDAN)
2. Connector stop with the component number, for which the connector stop is defined, set to 2 (CARTESIAN) or 6 (CARDAN)

Model: The models consist of a series of independent, 2-node connector elements with relevant connector behaviors.

Results and discussion

Abaqus matches the analytical solution for all relevant settings.

Input files

Abaqus/Standard input files

misc_lock_std_conn3d.inp	Lock connector behavior.
misc_stop_std_conn3d.inp	Stop connector behavior.

Abaqus/Explicit input files

misc_lock_xpl_conn3d.inp	Lock connector behavior.
misc_stop_xpl_conn3d.inp	Stop connector behavior.

Failure

Elements tested

CONN2D2 CONN3D2

Problem description

This section focuses on connector failure behavior. Both CARTESIAN and CARDAN connections are employed in these verification cases.

The behavior option is verified by applying a concentrated load or displacement such that the connector failure limits are exceeded.

For CARTESIAN connections, the following connector failure cases are tested:

1. Connector failure with the connector's component number for which a failure criterion is defined set to 1 and all available components of relative motions released when the failure criterion is satisfied
2. Connector failure with the connector's component number for which a failure criterion is defined set to 3 and the available component of relative motion number to release, when the failure criterion is satisfied, set to 2

For CARDAN connections, the following connector failure cases are tested:

1. Connector failure with the connector's component number for which a failure criterion is defined set to 4 and all available components of relative motion released when the failure criterion is satisfied
2. Connector failure with the connector's component number for which a failure criterion is defined, set to 6, and the available component of relative motion number to release, when the failure criterion is satisfied, set to 4

Model: The models consist of a series of independent, 2-node connector elements with relevant connector behaviors.

Results and discussion

Abaqus matches the expected solutions for all cases.

Input files

Abaqus/Standard input files

misc_fail_std_conn2d.inp	Failure connector behavior.
misc_fail_std_conn3d.inp	Failure connector behavior.

Abaqus/Explicit input files

misc_fail_xpl_conn2d.inp	Failure connector behavior.
misc_fail_xpl_conn3d.inp	Failure connector behavior.

Friction

Elements tested

CONN2D2 CONN3D2

Problem description

This section focuses on Coulomb-like friction behaviors using the connector friction, connector derived components, and connector potentials. Most connection types for which friction can be defined are tested, including: AXIAL, CARTESIAN, RADIAL-THRUST, SLIDE-PLANE, SLOT, CARDAN, EULER, FLEXION-TORSION, ROTATION, REVOLUTE, UNIVERSAL, CYLINDRICAL, HINGE, PLANAR, TRANSLATOR, and UJOINT.

The behavior options are verified by applying concentrated loads or displacements to create nonzero contact forces and some relative motion in the connectors. The friction-related output quantities (friction forces, contact forces, and relative slip) are monitored to assess the solution quality. In the Abaqus/Standard tests both static and direct-integration implicit dynamic procedures are performed. In many of the Abaqus/Standard input files, perturbation procedures (steady-state dynamics, frequency, and random response) are also performed with or without the load case definition. Both the predefined and the user-customized friction behavior are tested. Various friction models as defined by the friction coefficient specified by the contact surface interaction or by changes to friction properties.

Model: The models consist of a series of independent, 2-node connector elements with relevant connector behaviors.

Results and discussion

Abaqus matches the expected solutions for all cases.

Input files

Abaqus/Standard input files

misc_connfric_std_axial.inp	Friction connector behavior in the AXIAL connection type (three-dimensional test).
misc_connfric_std_axial2d.inp	Friction connector behavior in the AXIAL connection type (two-dimensional test).
misc_connfric_std_ballinsocket.inp	Friction connector behavior in the ball-in-socket-like connection types (JOIN and one of the following: CARDAN, FLEXION-TORSION, or ROTATION).
misc_connfric_std_ballconst.inp	Friction connector behavior in the ball-in-socket-like connection types (JOIN and one of the following: CARDAN, EULER, FLEXION-TORSION, or ROTATION) with constant contact force.
misc_connfric_std_bushing.inp	Friction connector behavior in the bushing-like connection types (modeled here with CARTESIAN and FLEXION-TORSION).
misc_connfric_std_cartesian.inp	Friction connector behavior in the CARTESIAN connection type (three-dimensional test).
misc_connfric_std_cartesian2d.inp	Friction connector behavior in the CARTESIAN connection type (two-dimensional test).
misc_connfric_std_cart2d_lock.inp	Friction connector behavior in the CARTESIAN connection type with **CONNECTOR LOCK* (two-dimensional test).
misc_connfric_std_cylindrical.inp	Friction connector behavior in the CYLINDRICAL connection type.
misc_connfric_std_hard.inp	Friction connector behavior in the HINGE and CYLINDRICAL connection types modeling hardening plasticity-like behavior.
misc_connfric_std_hinge.inp	Friction connector behavior in the HINGE connection type.
misc_connfric_std_planar.inp	Friction connector behavior in the PLANAR connection type.
misc_connfric_std_radialthrust.inp	Friction connector behavior in the RADIAL-THRUST connection type.
misc_connfric_std_revolute.inp	Friction connector behavior in the REVOLUTE connection type.
misc_connfric_std_slideplane.inp	Friction connector behavior in the SLIDE-PLANE connection type.
misc_connfric_std_slot.inp	Friction connector behavior in the SLOT connection type (three-dimensional test).
misc_connfric_std_slot_res.inp	Friction connector behavior in the SLOT connection type in a restart analysis.
misc_connfric_std_slot2d.inp	Friction connector behavior in the SLOT connection type (two-dimensional test).
misc_connfric_std_translator.inp	Friction connector behavior in the TRANSLATOR connection type.
misc_connfric_std_ujoint.inp	Friction connector behavior in the UJOINT connection type.

Abaqus/Explicit input files

misc_connfric_xpl_axial.inp	Friction connector behavior in the AXIAL connection type (three-dimensional test).
misc_connfric_xpl_axial2d.inp	Friction connector behavior in the AXIAL connection type (two-dimensional test).

misc_connfric_xpl_ballinsocket.inp	Friction connector behavior in the ball-in-socket-like connection types (JOIN and one of the following: CARDAN, FLEXION-TORSION, or ROTATION).
misc_connfric_xpl_ballconst.inp	Friction connector behavior in the ball-in-socket-like connection types (JOIN and one of the following: CARDAN, EULER, FLEXION-TORSION, or ROTATION) with constant contact force.
misc_connfric_xpl_bushing.inp	Friction connector behavior in the bushing-like connection types (modeled here with CARTESIAN and FLEXION-TORSION).
misc_connfric_xpl_cartesian.inp	Friction connector behavior in the CARTESIAN connection type (three-dimensional test).
misc_connfric_xpl_cartesian2d.inp	Friction connector behavior in the CARTESIAN connection type (two-dimensional test).
misc_connfric_xpl_cylindrical.inp	Friction connector behavior in the CYLINDRICAL connection type.
misc_connfric_xpl_hard.inp	Friction connector behavior in the HINGE and CYLINDRICAL connection types modeling hardening plasticity-like behavior.
misc_connfric_xpl_hinge.inp	Friction connector behavior in the HINGE connection type.
misc_connfric_xpl_planar.inp	Friction connector behavior in the PLANAR connection type.
misc_connfric_xpl_revolute.inp	Friction connector behavior in the REVOLUTE connection type.
misc_connfric_xpl_slideplane.inp	Friction connector behavior in the SLIDE-PLANE connection type.
misc_connfric_xpl_slot.inp	Friction connector behavior in the SLOT connection type (three-dimensional test).
misc_connfric_xpl_slot_res.inp	Friction connector behavior in the SLOT connection type in a restart analysis.
misc_connfric_xpl_slot2d.inp	Friction connector behavior in the SLOT connection type (two-dimensional test).
misc_connfric_xpl_translator.inp	Friction connector behavior in the TRANSLATOR connection type.
misc_connfric_xpl_ujoint.inp	Friction connector behavior in the UJOINT connection type.

Connector actuation options

Elements tested

CONN2D2 CONN3D2

Problem description

This section focuses on actuation behaviors using the available connector components of relative motion. CARTESIAN and CARDAN connections are used in these verification cases.

The available connector components of relative motion, with the values of variables being prescribed fixed at their values at the start of the step, are verified by inducing a relative displacement between the connector nodes in the first step of the load history, then fixing the motion and applying a concentrated load to verify no motion occurs.

The available connector components of relative motion, with a velocity or acceleration history specified, are verified by applying a relative velocity or acceleration to the connector element and obtaining a resulting relative displacement and connector load that correspond to the analytical solution for the prescribed conditions.

The available connector components of relative motion, with any nonzero magnitudes associated with prescribed variables defined in user subroutine *DISP*, are verified by applying a relative displacement between the connector nodes.

In the Abaqus/Standard tests a linear perturbation static procedure is performed in the last step.

Each of the cases listed above is tested.

Model: The models consist of a series of independent, 2-node connector elements with relevant connector behaviors.

Results and discussion

Abaqus matches the expected solutions for all cases.

Input files

Abaqus/Standard input files

misc_motionu_std_conn3d.inp	Connector displacement actuation.
misc_motionv_std_conn3d.inp	Connector velocity actuation.
misc_motiona_std_conn3d.inp	Connector acceleration actuation.

Abaqus/Explicit input files

misc_motionu_xpl_conn3d.inp	Connector displacement actuation.
misc_motionv_xpl_conn3d.inp	Connector velocity actuation.
misc_motiona_xpl_conn3d.inp	Connector acceleration actuation.

Restart, model change, and postprocessing analysis procedure

Elements tested

CONN2D2 CONN3D2

Problem description

These verification cases test connector elements with options not routinely tested in other verification problems: restart, element pair reactivation, and postprocessing analysis procedures.

Results and discussion

The analyses match the expected behavior.

Input files

misc_restart_std_conn3d.inp	**RESTART* with connector elements in Abaqus/Standard.
misc_dam_restart_std.inp	**RESTART* with connector damage in Abaqus/Standard.
misc_mdlch_std_conn3d.inp	**MODEL CHANGE* with connector elements in Abaqus/Standard.
misc_postout_std_conn3d.inp	**POST OUTPUT* with connector elements in Abaqus/Standard.

Connector elements in perturbation analyses

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Eigenvalue buckling analysis

Elements tested

CONN2D2 CONN3D2

Problem description

These verification cases test the performance of connector elements in eigenvalue buckling procedures. AXIAL, CARTESIAN, and CARDAN connections with elastic connector behavior are employed. Elastic connector behavior is defined with the connector elasticity procedure. Perturbation loads are applied via connector actuation using both the connector element load and available connector components of relative motion procedures. When the load is applied with available connector components of relative motion, the connector motion for the application of loads or the connector motion for the buckling modes can be defined. Results are verified by comparison with either analytical solutions or numerical results from equivalent models without connector elements.

Model: The models consist of a series of 2-node connector elements that support and actuate a column. The column is modeled with beam elements.

Results and discussion

Abaqus results match the expected solution for all cases.

Input files

buckle_conn2d.inp	Eigenvalue buckling analysis with CONN2D2 elements.
buckle_conn3d.inp	Eigenvalue buckling analysis with CONN3D2 elements.

Natural frequency extraction

Elements tested

CONN2D2 CONN3D2

Problem description

These verification cases test the performance of connector elements in natural frequency extraction procedures. AXIAL, CARTESIAN, and CARDAN connections with elastic connector behavior are employed. Elastic connector behaviors are defined. Results are verified by comparison with either analytical solutions or numerical results from equivalent models without connector elements.

Model: The models consist of a series of independent, 2-node connector elements that support and actuate a column. The column is modeled with beam elements.

Results and discussion

Abaqus results match the expected solution for all cases.

Input files

freq_conn2d.inp	Frequency extraction analysis with CONN2D2 elements.
freq_conn3d.inp	Frequency extraction analysis with CONN3D2 elements.

Transient modal dynamic analysis

Elements tested
CONN2D2 CONN3D2

Problem description

These verification cases test the performance of connector elements in transient modal dynamic procedures. AXIAL, CARTESIAN, and CARDAN connections with elastic connector behavior are employed. Elastic connector behavior is defined. Results are verified by comparison with either analytical solutions or numerical results from equivalent models without connector elements.

Model: The models consist of a series of 2-node connector elements supporting a column that is subjected to a dynamic load. The column is modeled with beam elements.

Results and discussion

Abaqus results match the expected solution for all cases.

Input files

modal_conn2d.inp	Transient modal dynamic analysis with CONN2D2 elements.
modal_conn3d.inp	Transient modal dynamic analysis with CONN3D2 elements.

Steady-state dynamic analyses: Direct, modal, and subspace

Elements tested
CONN2D2 CONN3D2

Problem description

These verification cases test the performance of connector elements in steady-state dynamic analyses. Abaqus offers the direct-solution steady-state procedure and the modal based steady-state dynamic and subspace-based steady-state dynamic procedures. The connection types AXIAL, ROTATION, CARTESIAN, and CARDAN are tested in these procedures. Elastic and damping connector behaviors are defined for all connections. Results are verified by comparison with either analytical solutions or numerical results from equivalent models without connector elements.

Model: The models consist of three connector elements with nodal masses. Two connector elements are connected in series and actuated by the third connector. Actuation is achieved by specifying the connector element loads of the relative motion of available components. Both the real and imaginary parts of the loading are specified.

Results and discussion

Abaqus results match the expected solution for all cases.

Input files

ssd_conn2d_axi.inp	Steady-state dynamics, AXIAL connectors.
ssd_conn2d_rot.inp	Steady-state dynamics, ROTATION connectors.
ssd_conn3d_cart.inp	Steady-state dynamics, CARTESIAN connectors, perturbation step with **LOAD CASE*.
ssd_conn3d_cardan.inp	Steady-state dynamics, CARDAN connectors.

Response spectrum analysis

Elements tested
CONN2D2 CONN3D2

Problem description

These verification cases test the performance of connector elements in response spectrum analysis. Both AXIAL and CARTESIAN connections are employed. Elastic and damping connector behaviors are defined for the connections. Results are verified by comparison with either analytical solutions or numerical results from equivalent models without connector elements.

Model: The models consist of three connector elements with nodal masses. The system is subjected to both a displacement and a velocity spectrum.

Results and discussion

Abaqus results match the expected solution for all cases.

Input files

rs_conn2d_axi.inp	Response spectrum analysis, AXIAL connectors.
rs_conn3d_cart.inp	Response spectrum analysis, CARTESIAN connectors.
conn_quake_dis.inp	Input data for the displacement spectrum.
conn_quake_vel.inp	Input data for the velocity spectrum.

Random response analysis

Elements tested
CONN2D2 CONN3D2

Problem description

These verification cases test the performance of connector elements in random response analysis. AXIAL, ROTATION, CARTESIAN, and CARDAN connections are employed. Elastic and damping connector behaviors are defined for the connections. The system is exposed to a nondeterministic loading applied via the connector element load procedure. The cross-spectral density frequency function of the random loading is specified with the power spectral density definition. The case considered here is uncorrelated white noise. Results are verified

by comparison with either analytical solutions or numerical results from equivalent models without connector elements.

Model: The models consist of three connector elements with nodal masses. Two connector elements are connected in series and actuated by the third connector with a nondeterministic load.

Results and discussion

Abaqus results match the expected solution for all cases.

Input files

random_conn2d_axi.inp	Random response analysis, AXIAL connectors.
random_conn2d_rot.inp	Random response analysis, ROTATION connectors.
random_conn3d_cart.inp	Random response analysis, CARTESIAN connectors.
random_conn3d_cardan.inp	Random response analysis, CARDAN connectors.

Connector lock, stop, plasticity, damage, and friction behavior in perturbation procedures

Elements tested

CONN2D2 CONN3D2

Problem description

These verification cases test the performance of lock, stop, plasticity, damage, and friction connector behaviors in perturbation analyses, defined with the connector lock, connector stop, connector plasticity, and connector hardening, connector damage initiation and connector damage evolution, and connector friction behaviors, respectively. These options are tested separately. Both AXIAL and CARDAN connections are employed.

Plastic relative motions do not change in linear perturbation procedures. Frictional slipping is not allowed during linear perturbation procedures; thus, all available components of relative motion with connector friction behavior should remain fixed and equal to the values from the base state. Similarly, the status of connector locks and stops cannot change during a linear perturbation analysis. The performance of lock, stop, plasticity, and friction connector behavior is tested in both the frequency and the direct-solution steady-state dynamic procedures. The behavior options are verified through a multistep load history. The perturbation steps are preceded by general static steps where a load is applied such that the corresponding prescribed limits for the locking, stopping, plasticity, damage initiation, or friction behavior are exceeded. For the lock and stop cases the load direction is reversed in a subsequent step to confirm the locking or stopping behavior.

Model: The models consist of three connector elements with nodal masses. One of the connectors has the relevant lock, stop, plasticity, damage, or friction behaviors.

Results and discussion

Abaqus results match the expected solution for all cases.

Input files

lock_conn2d_axi.inp	Lock connector behavior, AXIAL connectors.
lock_conn3d_cardan.inp	Lock connector behavior, CARDAN connectors.
stop_conn2d_axi.inp	Stop connector behavior, AXIAL connectors.

stop_conn3d_cardan.inp	Stop connector behavior, CARDAN connectors.
plasdam_conn2d_axi.inp	Plasticity connector behavior, AXIAL connectors, perturbation step with **LOAD CASE*.
plasdam_conn3d_cardan.inp	Plasticity connector behavior, CARDAN connector.
damage_conn3d_cardan.inp	Damage connector behavior, CARDAN connector.
frict_conn2d_axi.inp	Friction connector behavior, AXIAL connectors, perturbation step with **LOAD CASE*.
frict_conn3d_cardan.inp	Friction connector behavior, CARDAN connectors.

Tests for special-purpose connectors

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Frictionless SLIPRING connectors

Elements tested

CONN3D2

Problem description

The SLIPRING connection type is verified via a frictionless pulley and inextensible belt system. Results are compared against well-known analytical results.

A high elastic modulus is specified for the belt of the SLIPRING via the connector elasticity procedure to achieve inextensible behavior. The analysis compares the results of two separate pulley-belt systems, each displacing similar loads though the same distance. Each system models the belt passing over the pulley using two SLIPRING connector elements sharing a common node. A load of 10 units is applied at the common node of the SLIPRING-type connector elements. In each system one of the ends of the belt is fully fixed, and different sets of boundary conditions are applied at the other free end to displace the applied load by similar distance, as described below.

1. System 1:
   a. Apply boundary conditions to constrain degrees of freedom 1, 2, 3, and 10 (the material flow degree of freedom) at the left end of the belt system.
   b. Apply boundary conditions at the right end to constrain degrees of freedom 1, 2, and 3.
   c. Apply velocity-type boundary conditions on degree of freedom 10 at the free end (to pull out 2.25 units of belt material).
2. System 2:
   a. Apply boundary conditions to constrain degrees of freedom 1, 2, 3, and 10 at the left end of the belt system.
   b. Apply boundary conditions at the right end to constrain degrees of freedom 1, 3, and 10.
   c. Apply velocity-type boundary conditions on degree of freedom 2 at the free end (to displace the node by 2.25 units).

Results and discussion

The load displaces by 1.125 units in both systems, which is half the 2.25 unit length of belt material pulled out at the free end, a well-known analytical result. The belt tension in each SLIPRING is 5 units, half the 10 unit load applied to each system and again matching analytical results. In all cases the material flowing out of the first connector element equals the material flowing into the second connector element.

Input files

Abaqus/Standard input file

misc_elasslipring_std_conn3d.inp SLIPRING with linear elastic connector behavior.

Abaqus/Explicit input file

misc_elasslipring_xpl_conn3d.inp SLIPRING with linear elastic connector behavior.

Frictional SLIPRING connectors

Elements tested

CONN3D2

Problem description

Frictional behavior in the SLIPRING connection type is verified by comparing computed results with the analytical reference solution. Both linear elastic and nonlinear elastic connector behaviors have been verified in separate tests. The test consists of a system of two pulleys and a belt passing over the pulleys, which is modeled using three SLIPRING connections. The angle between adjacent SLIPRING connections is held constant at 90°. Concentrated nodal loads are applied at the two free ends. A time varying amplitude is specified for these loads to cause the belt to slip in one direction first and then reverse and slip in the opposite direction. The coefficient of friction μ is 0.1.

Frictional contact behavior in the SLIPRING connection type is also verified for a case with a varying contact angle between the belt and the pulley. *Figure 1* depicts a belt-pulley system between nodes 1, 2 and 3. The belt is modeled using two slipring connector elements and passes over a pulley at node 2. The frictional force between the belt and the pulley depends on the contact angle between them. The model is setup with two steps. In the first step the contact angle remains constant as a concentrated load of magnitude 1×10^5 is applied to pull node 1 down along Y-direction, while nodes 2 and 3 are held fixed. In the second step, the contact angle varies as node 3 is displaced along the X-direction. The concentrated load applied on node 1 is held constant during the second step. The coefficient of friction μ is assumed to be 0.1.

Results and discussion

When the belt slips, the ratio of the belt tensions in the adjacent SLIPRING connections is given by $t_2 = t_1 e^{\mu\alpha}$ when $t_2 > t_1$ and by $t_2 = t_1 e^{-\mu\alpha}$ when $t_2 < t_1$. For the tests with constant friction angle, it is verified that for linear and nonlinear elastic behavior, the belt tension ratio changes from $e^{0.1\ \ 0.5\ \ \pi} = 1.17$ to $e^{-0.1\ \ 0.5\ \ \pi} = 0.854$ as the belt reverses in slip direction.

Figure 2 shows a plot of the belt tension force versus the horizontal displacement of node 3 for the varying contact angle case. The contact angle changes from $5\pi/6$ at the beginning of the second step to $2\pi/3$ at the end. As node 3 is displaced in the X-direction, the slip direction between the belt and the pulley eventually reverses direction. This corresponds to a flip in the tension force in belt segment 2–3 from a value of $10^5 \exp(-0.1\ \ 5\pi/6) = 7697$ to a value of $10^5 \exp(0.1\ \ 5\pi/6) = 12993$. Subsequently, the belt tension force decreases with the change in the contact angle to a final value of $10^5 \exp(0.1\ \ 2\pi/3) = 12330$.

Input files

Abaqus/Standard input files

misc_slipring3_fric_conn3d_std.inp	Linear elastic connector behavior with connector friction.
misc_slipring3_fric_conn2d_std.inp	Linear elastic connector behavior with connector friction.
misc_slipring_nlelasplas_conn3d_std.inp	Nonlinear elastic-plastic connector behavior with connector friction.
SlipringUpdatedAngle.inp	Linear elastic connector behavior with connector friction and varying contact angle.

Abaqus/Explicit input files

misc_slipring3_fric_conn3d_xpl.inp	Linear elastic connector behavior with connector friction.
misc_slipring3_nlelasfric_conn3d_xpl.inp	Nonlinear elastic connector behavior with connector friction.

RETRACTOR connectors

Elements tested

CONN2D2 CONN3D2

Problem description

These verification cases test the RETRACTOR (FLOW-CONVERTER) connection types. Two sets of RETRACTOR connections are used. In the first case the material flow degree of freedom (10) at node b is driven via boundary condition and the degree of freedom 6 is measured at node a (all other degrees of freedom at the nodes are held fixed). In the second case degree of freedom 6 at node a is driven via boundary condition and degree of freedom 10 is measured at node b (all other degrees of freedom at the nodes are held fixed).

Results and discussion

The measured material flow and rotations agree with the applied boundary conditions.

Input files

Abaqus/Standard input files

misc_flowconverter_std_conn3d.inp	Retractor-type connection.
misc_flowconverter_std_conn2d.inp	Retractor-type connection.
misc_slipringretractor_conn3d_std.inp	SLIPRING- and retractor-type connection.

Abaqus/Explicit input file

misc_flowconverter_xpl_conn3d.inp	Retractor-type connection.

ACCELEROMETER connectors

Elements tested

CONN3D2

Problem description

These verification cases test the ACCELEROMETER and ROTATION-ACCELEROMETER connection types. In the first case an ACCELEROMETER connection is used in conjunction with a BEAM connector. Node 1 of the BEAM connector is constrained in degrees of freedom 1, 2, and 3. Node 1 of the accelerometer is fully constrained, and node 2 of the accelerometer is constrained to move radially by the BEAM connector. Node 2 of the accelerometer is moved via the connector motion procedure, with a velocity history of magnitude V in the local 2 direction. The angular velocity at node 1 of the BEAM connector, about the axis of rotation, is V/R, where R is the length of the BEAM connector.

The configuration of the second case is identical to that in case 1. However, in this case in addition to an ACCELEROMETER, a ROTATION-ACCELEROMETER is also defined between the same two nodes. In this case an angular velocity of $\boldsymbol{\omega}$ is applied to node 1 of the BEAM connector. Node 2 of accelerometer moves along the radial path with a velocity of constant magnitude $\boldsymbol{V = R * \omega}$. Node 2 of the accelerometer is constrained to have the same angular velocity $\boldsymbol{\omega}$ since it is also node 2 of the BEAM connector.

Results and discussion

For case 1 the angular velocity at node 1 of the BEAM connector agrees with the applied velocity history of the accelerometer, specified via the connector motion procedure. For case 2 the translational velocity in the local system of the accelerometer agrees with the applied angular velocity at node 1 of the BEAM connector. The rotational velocity in the local system of the ROTATION-ACCELEROMETER also agrees with the applied angular velocity at node 1 of the BEAM connector.

Input files

Abaqus/Explicit input file

misc_acclmeter_xpl_conn3d.inp Accelerometer-type connection.

Figures

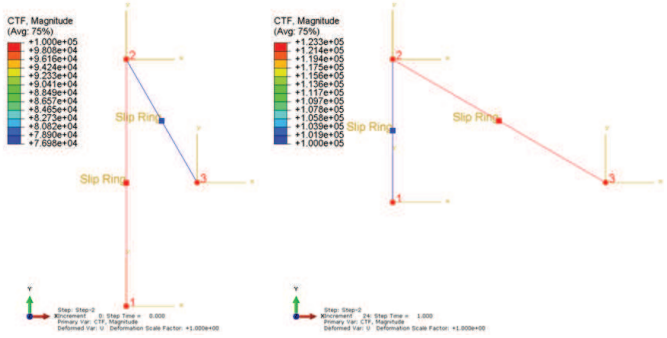


Figure 1: The initial and final configuration of the SLIPRING system.

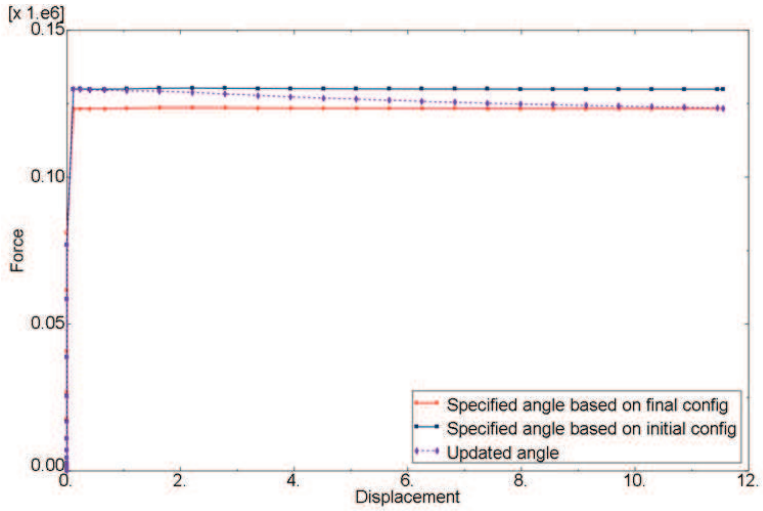


Figure 2: Reaction force versus displacement at node 3.

Special-purpose stress/displacement elements

In this section:

- *Flexible joint element*
- *Line spring elements*
- *Distributing coupling elements*
- *Drag chain elements*

Flexible joint element

Products: Abaqus/Standard

Overview

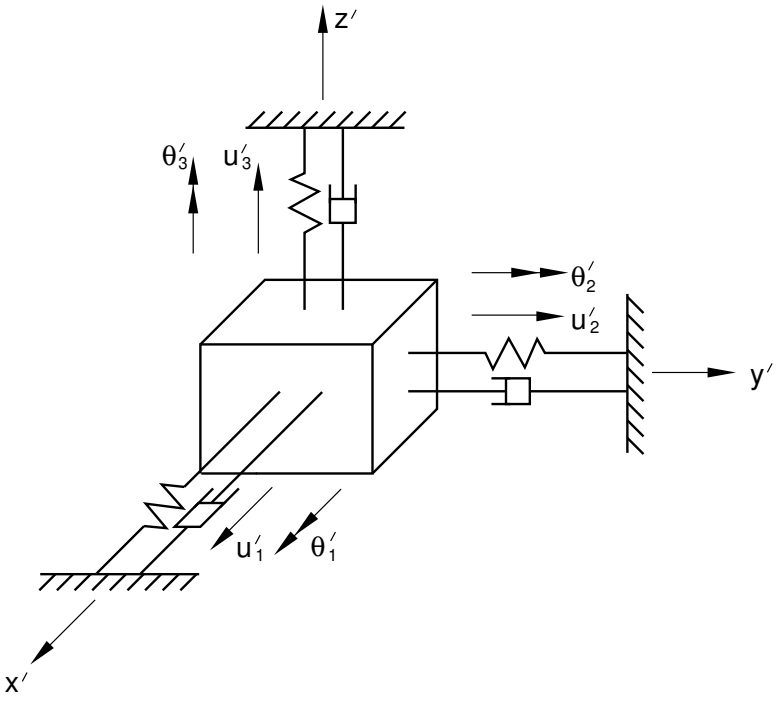
This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

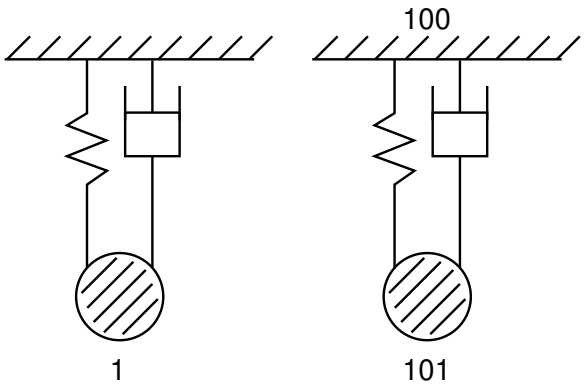
JOINTC

Problem description

The behavior of the joint is defined in a local coordinate system that rotates with the motion of the first node of the JOINTC element. The first three tests consist of linear springs that couple the corresponding components of relative displacement and of relative rotation in the joint.



The fourth test includes linear dashpots. A spring and dashpot system is modeled using SPRING1 and DASHPOT1 elements and also with a JOINTC element utilizing the dashpot coefficient.



Material properties used for the first three tests: Linear elastic; spring stiffnesses for relative displacements are 100, 200, and 300 for degrees of freedom 1, 2, and 3, respectively; spring stiffnesses for relative rotations are 400, 500, and 600 for degrees of freedom 4, 5, and 6, respectively.

Material properties used for the fourth test: Linear elastic, spring stiffnesses = 30.0 for degree of freedom 1, dashpot coefficients = 0.12 for degree of freedom 1, mass = 0.02588 at node 1.

Boundary conditions for linear behavior: Node 1 is clamped.

Loading for linear behavior: Step 1: Displacements at node 2 are prescribed to 1.0×10^{-3} for all degrees of freedom.

Step 2: Applied forces and moments at node 2 are equal to 1.0 for all components.

Boundary conditions and loading for nonlinear behavior with a specified local coordinate system: Step 1: Node 1 is clamped. $F_x = 100$ at node 2.

Step 2: A rotation of 90° is prescribed about the global 3-axis at node 1 (see (*) below).

Boundary conditions and loading for nonlinear behavior with applied rotations and moments: Step 1: Node 1 is clamped. A moment of magnitude 80 is applied about the global 1-axis at node 2.

Step 2: The moment is removed.

Step 3: A rotation of 90° is prescribed about the global 3-axis at node 1. All other degrees of freedom at node 1 are suppressed.

Step 4: In addition to the conditions at the end of the previous step, a moment of magnitude 80 is applied about the global 2-axis at node 2.

Boundary conditions and loading for linear behavior with the dashpot coefficient:
Step 1 (static): Node 100 is clamped, node 101 has $u_1 = 1.0$ and all other degrees of freedom suppressed, node 1 has $u_1 = 1.0$.

Step 2 (dynamic): The applied displacements at nodes 1 and 101 are released.

Results and discussion

The results for each test are tabulated and discussed below.

Linear behavior

Table 1: Displacements at node 2.

Step	u_1	u_2	u_3	ϕ_1	ϕ_2	ϕ_3
1	1.0×10^{-3}	1.0×10^{-3}	1.0×10^{-3}	1.0×10^{-3}	1.0×10^{-3}	1.0×10^{-3}
2	1.0×10^{-2}	5.0×10^{-3}	3.33×10^{-3}	2.5×10^{-3}	2.0×10^{-3}	1.67×10^{-3}

Table 2: Reactions at node 1.

Step	RF_1	RF_2	RF_3	RM_1	RM_2	RM_3
1	–0.1	–0.2	–0.3	–0.4	–0.5	–0.6
2	–1.0	–1.0	–1.0	–1.0	–1.0	–1.0

Table 3: Element strains.

Step	E_{11}	E_{22}	E_{33}	E_{12}	E_{13}	E_{23}
1	1.0×10^{-3}	1.0×10^{-3}	1.0×10^{-3}	1.0×10^{-3}	1.0×10^{-3}	1.0×10^{-3}
2	1.0×10^{-2}	5.0×10^{-3}	3.33×10^{-3}	2.5×10^{-3}	2.0×10^{-3}	1.67×10^{-3}

Table 4: Element stresses.

Step	S_{11}	S_{22}	S_{33}	S_{12}	S_{13}	S_{23}
1	0.1	0.2	0.3	0.4	0.5	0.6
2	1.0	1.0	1.0	1.0	1.0	1.0

Nonlinear behavior with the local coordinate system procedure

Table 5: Displacements at node 2.

ϕ_z (*)	u_1	u_2	u_3	ϕ_1	ϕ_2	ϕ_3
0.0	1.0	0.0	0.0	0.0	0.0	0.0
30.0	0.875	0.217	0.0	0.0	0.0	0.524
90.0	0.5	1.34×10^{-8}	0.0	0.0	0.0	1.571

Table 6: Reactions at node 1.

ϕ_z (*)	RF_1	RF_2	RF_3	RM_1	RM_2	RM_3
0.0	–100.0	0.0	0.0	0.0	0.0	0.0
30.0	–100.0	0.0	0.0	0.0	0.0	21.65
90.0	–100.0	0.0	0.0	0.0	0.0	1.34×10^{-6}

(*) Prescribed rotation at node 1: $\phi_z = 0$ at the end of Step 1; $\phi_z = 30$ at Step 2, increment 3; $\phi_z = 90$ at Step 2, increment 9.

Nonlinear behavior with applied rotations and moments

Table 7: Displacements at node 2.

Step	Inc.	ϕ_1	ϕ_2	ϕ_3
1	1	0.1007	0.0	0.0
1	2	0.2058	0.0	0.0
4	1	7.91×10^{-2}	7.91×10^{-2}	1.569
4	2	0.1616	0.1616	1.565

Table 8: Reactions at node 1.

Step	Inc.	RM_1	RM_2	RM_3
1	1	−40.0	0.0	0.0
1	2	−80.0	0.0	0.0
4	1	0.0	−40.0	0.0
4	2	0.0	−80.0	0.0

Table 9: Element strains.

Step	Inc.	E_{12}	E_{13}	E_{23}
1	1	0.1005	0.0	0.0
1	2	0.2043	0.0	0.0
4	1	0.1005	0.0	0.0
4	2	0.2043	0.0	0.0

Table 10: Element stresses.

Step	Inc.	S_{12}	S_{13}	S_{23}
1	1	40.20	0.0	0.0
1	2	81.72	0.0	0.0
4	1	40.20	1.08×10^{-6}	-1.09×10^{-8}
4	2	81.72	2.18×10^{-6}	-4.55×10^{-8}

Linear behavior with the dashpot coefficient procedure

The displacement histories of nodes 1 and 101 match.

Input files

exjoxlx1.inp	Linear behavior.
exjoxox1.inp	Nonlinear behavior with the local coordinate system procedure.
exjoxrx1.inp	Nonlinear behavior with applied rotations and moments.

exjoxdx1.inp	Linear behavior with the dashpot coefficient procedure.

Input files *exjoxlxa.inp*, *exjoxoxa.inp*, *exjoxrxa.inp*, and *exjoxdxa.inp* are modified versions of files *exjoxlx1.inp*, *exjoxox1.inp*, *exjoxrx1.inp*, and *exjoxdx1.inp*, respectively. They include temperature- and/or field variable-dependent behavior for spring constants and dashpot coefficients where applicable. These modified files are designed to provide exactly the same results as those files from which they are derived.

Line spring elements

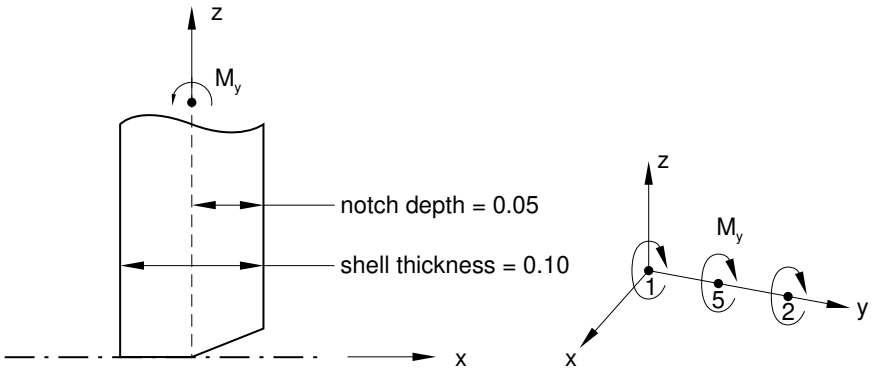
Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

LS3S with constant-depth notch under far-field bending

Problem description




```
*ELEMENT, TYPE=LS3S, ELSET=ALL
1, 2, 5, 1
*SURFACE FLAW, SIDE=POSITIVE
1, .05
5, .05
2, .05
*SHELL SECTION, MAT=M1, ELSET=ALL
.1,
```

Material: Linear elastic, Young's modulus = 1.0, Poisson's ratio = 0.0.

Boundary conditions: $u_x = u_y =$ 0 at nodes 1, 2, and 5.

Loading: $M_y = -1.0$ at nodes 1 and 2. $M_y = -4.0$ at node 5.

Results and discussion

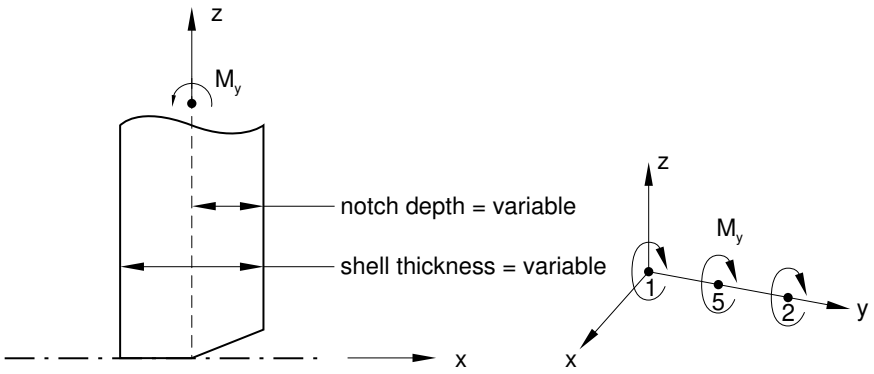
Element	Pt.	J	K	$J_{elastic}$	$J_{plastic}$
1	1	4.43×10^6	2105.0	4.43×10^6	0.0
1	2	4.43×10^6	2105.0	4.43×10^6	0.0
1	3	4.43×10^6	2105.0	4.43×10^6	0.0

Input files

exls3bx2.inp Single-edge constant-depth notch strip under far-field bending.

LS3S with variable depth notch under far-field bending

Problem description




```
*ELEMENT, TYPE=LS3S, ELSET=ALL
1, 2, 5, 1
*SURFACE FLAW, SIDE=POSITIVE
1, .07
5, .05
2, .04
*SHELL SECTION, MAT=M1, ELSET=ALL, NODAL THICKNESS
99,
*NODAL THICKNESS
1, 0.7
5, 0.5
2, 0.4
3, 0.1
4, 0.1
6, 0.1
7, 0.1
8, 0.1
```

Material: Linear elastic, Young's modulus = 1.0, Poisson's ratio = 0.0.

Boundary conditions: $u_x = u_y =$ 0 at nodes 1, 2, and 5.

Loading: $M_y = -1.0$ at nodes 1 and 2. $M_y = -4.0$ at node 5.

Results and discussion

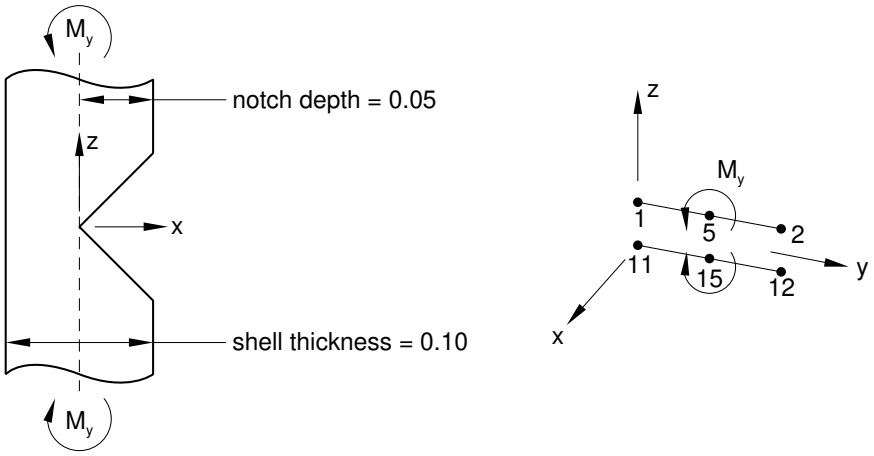
Element	Pt.	J	K	$J_{elastic}$	$J_{plastic}$
1	1	6891.0	83.012	6891.0	0.0
1	2	3528.5	59.401	3528.5	0.0
1	3	1286.2	35.864	1286.2	0.0

Input files

exls3vx2.inp Single-edge variable-depth notch strip under far-field bending.

LS6 under far-field bending

Problem description




```
*ELEMENT, TYPE=LS6, ELSET=ALL
1, 2, 5, 1, 12, 15, 11
*SURFACE FLAW, SIDE=POSITIVE
1, .05
5, .05
2, .05
*SHELL SECTION, MAT=M1, ELSET=ALL
.1,
```

Material: Linear elastic, Young's modulus = 1.0, Poisson's ratio = 0.0.

Boundary conditions: Node 17 is fully constrained. $\phi_z = 0$ for all nodes. Nodes 1, 2, and 5 are constrained to move together. Nodes 11, 12, and 15 are constrained to move together.

Loading: $M_y = -6.0$ at node 5. $M_y = 6.0$ at node 15.

Results and discussion

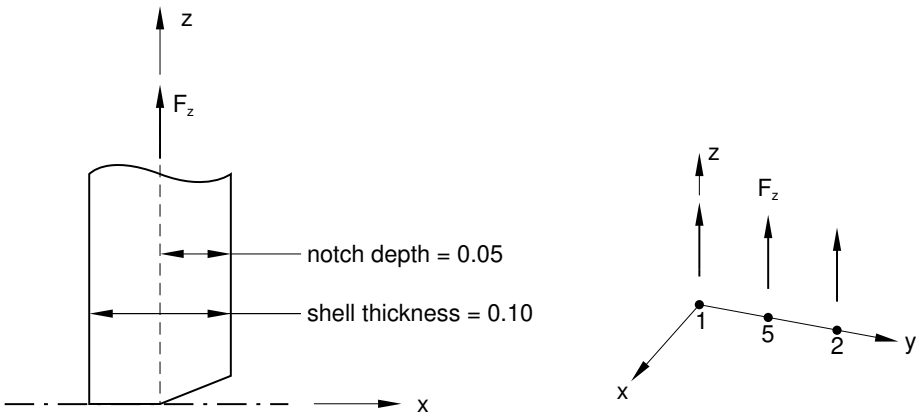
Element	Pt.	J	$J_{elastic}$	$J_{plastic}$	K_I	K_{II}	K_{III}
1	1	4.43×10^6	4.43×10^6	0.0	2105.0	0.0	0.0
1	2	4.43×10^6	4.43×10^6	0.0	2105.0	0.0	0.0
1	3	4.43×10^6	4.43×10^6	0.0	2105.0	0.0	0.0

Input files

exls6bx2.inp Single-edge notch strip under far-field bending about an axis (along the crack-tip line).

LS3S under far-field tension

Problem description




```
*ELEMENT, TYPE=LS3S, ELSET=ALL
1, 2, 5, 1
*SURFACE FLAW, SIDE=POSITIVE
1, .05
5, .05
2, .05
*SHELL SECTION, MAT=M1, ELSET=ALL
.1,
```

Material: Linear elastic, Young's modulus = 1.0, Poisson's ratio = 0.0.

Boundary conditions: $u_x = u_y =$ 0 at nodes 1, 2, and 5.

Loading: F_z= 1.0 at nodes 3 and 4. F_z= 4.0 at node 7.

Results and discussion

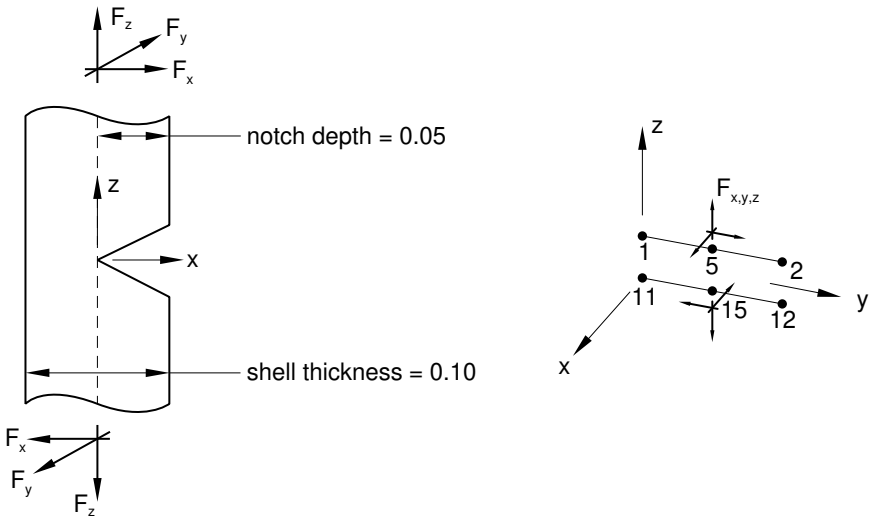
Element	Pt.	J	K	$J_{elastic}$	$J_{plastic}$
1	1	4518.0	67.22	4518.0	0.0
1	2	4518.0	67.22	4518.0	0.0
1	3	4518.0	67.22	4518.0	0.0

Input files

exls3tx2.inp Single-edge notch strip under far-field tension.

LS6 under Mode I, II, and III loading

Problem description




```
*ELEMENT, TYPE=LS6, ELSET=ALL
1, 2, 5, 1, 12, 15, 11
*SURFACE FLAW, SIDE=POSITIVE
1, .05
5, .05
2, .05
*SHELL SECTION, MAT=M1, ELSET=ALL
.1,
```

Material: Linear elastic, Young's modulus = 1.0, Poisson's ratio = 0.0.

Boundary conditions: Node 17 is fully constrained. $\phi_z = 0$ for all nodes. Nodes 1, 2, and 5 are constrained to move together. Nodes 11, 12, and 15 are constrained to move together.

Loading: $F_x = F_y = F_z =$ 1.0 at node 5. $F_x = F_y = F_z =$ −1.0 at node 15.

Results and discussion

Element	Pt.	J	$J_{elastic}$	$J_{plastic}$	K_I	K_{II}	K_{III}
1	1	170.10	170.10	0.0	11.20	4.962	−4.472
1	2	170.10	170.10	0.0	11.20	4.962	−4.472
1	3	170.10	170.10	0.0	11.20	4.962	−4.472

Input files

exls6sx2.inp Single-edge notch strip under far-field tension (Mode I), in-plane shear (Mode II), and uniform out-of-plane shear (Mode III).

Distributing coupling elements

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

DCOUP2D DCOUP3D

Problem description

The initial starting geometry for each test is shown in *Figure 1*. In the linear tests each coupling node is connected by a spring to ground (SPRING1) in each direction. In the geometrically nonlinear tests each coupling node is connected by a dashpot to ground (DASHPOT1) in each direction, and an axial spring element (SPRINGA) connects each pair of coupling nodes.

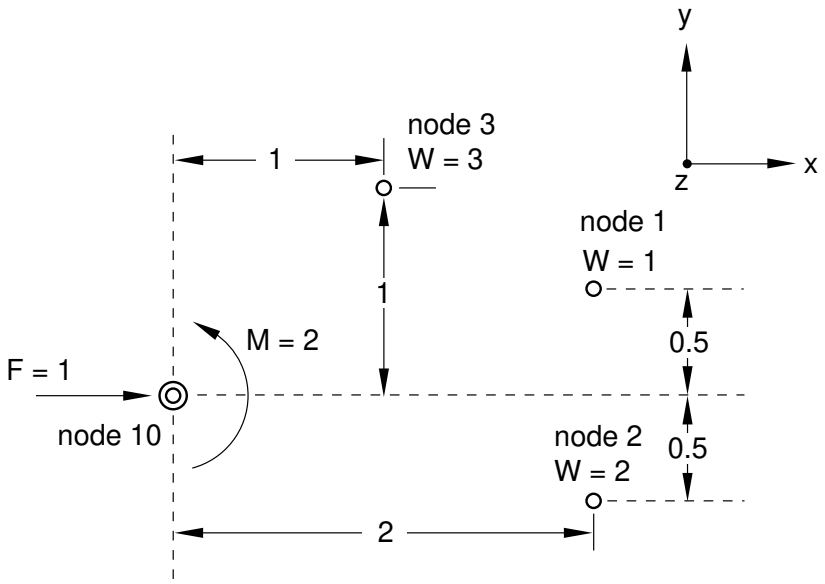
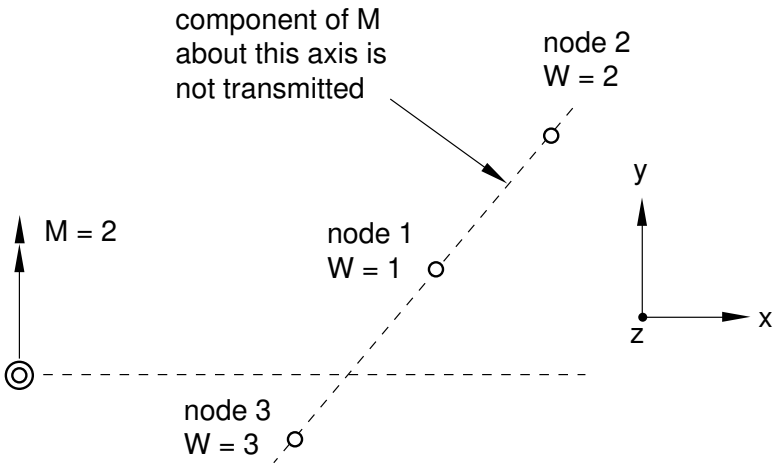


Figure 1: Initial starting geometry.

Distributing coupling elements connect a single reference node that has translational and rotational degrees of freedom to a collection of coupling nodes that have only translational degrees of freedom. Thus, when the coupling nodes are colinear, a situation can arise where the moments applied to the reference node are not transmitted by the element. This condition is relevant only for the three-dimensional version of the element. The third problem in this section tests the behavior of the element in this pathological situation.



Linear behavior

Properties:

The spring stiffnesses are 100, 200, and 300 for degrees of freedom 1, 2, and 3, respectively, for the springs connected to all coupling nodes. The mass of the distributing coupling is 10. The weight factors are 1, 2, and 3 for nodes 1, 2, and 3, respectively.

Loading:

Step 1: The force at node 10 is 1.0 in the x-direction. The moment at node 10 is 2.0 about the z-axis.

Step 2: (DCOUP3D only) The force at node 10 is 1.0 in the y-direction. The moment at node 10 is 2.0 about the x-axis.

Step 3: (DCOUP3D only) The force at node 10 is 1.0 in the z-direction. The moment at node 10 is 2.0 about the y-axis.

Step 4: Frequency extraction. (Step 2 for DCOUP2D)

Step 5: Transient modal dynamic step with a load, $F_x = 1.0 \sin 2\pi t$, applied to node 10. (Step 3 for DCOUP2D)

Step 6: Mode-based steady-state dynamic step with a load, $F_x = 1.0$, applied to node 10. (Step 4 for DCOUP2D)

Nonlinear behavior

Properties:

The dashpot damping coefficients are 100, 200, and 300 for degrees of freedom 1, 2, and 3, respectively, for the dashpots connected to all coupling nodes. The axial springs connecting the coupling nodes each have a spring constant of 1.0×10^8. The mass of the distributing coupling is 10.

Prescribed reference node motion:

Step 1: Total rotation of 2π about the z-axis. Translation $x = \sin 2\pi t$.

Step 2: (DCOUP3D only) Total rotation of 2π about the *y*-axis. Translation $z = \sin 2\pi t$.

Step 3: (DCOUP3D only) Total rotation of 2π about the *x*-axis. Translation $y = \sin 2\pi t$.

Step 4: Direct-integration dynamic step with a total rotation of 2π about the *x*-axis. Translation $x = \sin 2\pi t$. (Step 2 for DCOUP2D)

Behavior with a colinear arrangement

Properties:

The spring stiffnesses are 100, 200, and 300 for degrees of freedom 1, 2, and 3, respectively, for the springs connected to all coupling nodes. The total mass of the model is 10; and mass elements are defined at nodes 1, 2, and 3.

Loading:

Step 1: The moment at node 10 is 2.0 about the *z*-axis.

Step 2: The moment at node 10 is 2.0 about the *x*-axis.

Step 3: The moment at node 10 is 2.0 about the *y*-axis.

Step 4: The moment at node 10 has a magnitude of 2.0 and is parallel to the coupling node colinear axis.

Step 5: Frequency extraction.

Reference solution

In all tests the load distribution among coupling nodes adheres to the relation

$$\mathbf{F}^n = \widehat{w}^n \left(\mathbf{F}^R + \mathbf{T}^{-1} \left(\mathbf{M}^R + \mathbf{r}^R \times \mathbf{F}^R \right) \times \mathbf{r}^n \right),$$

where $\mathbf{F}^n$ is the force distribution at the coupling nodes, $\mathbf{F}^R$ and $\mathbf{M}^R$ are the force and moment at the reference node, $\widehat{w}^n$ are the normalized version of the weight factors specified with distributing coupling constraints, $\mathbf{T}$ is the coupling node arrangement inertia tensor, and $\mathbf{r}^R$ and $\mathbf{r}^n$ are the positions of the reference and coupling nodes relative to the coupling node arrangement centroid, respectively. See *Distributing coupling constraints* for a more detailed description of this load distribution.

Results and discussion

The results for each problem are discussed below.

Linear behavior

Table 1: Displacements at node 10.

Step	u_1	u_2	u_3
1	6.67×10^{-3}	-1.67×10^{-2}	0.0
2	-2.06×10^{-3}	1.35×10^{-2}	-2.67×10^{-2}
3	0.0	0.0	8.50×10^{-2}

Table 2: Rotations at node 10.

Step	ϕ_1	ϕ_2	ϕ_3
1	0.0	0.0	1.05×10^{-2}
2	1.33×10^{-2}	-1.33×10^{-2}	-7.33×10^{-3}
3	-2.67×10^{-2}	4.50×10^{-2}	0.0

Table 3: Displacements at node 1.

Step	u_1	u_2	u_3
1	1.19×10^{-3}	1.44×10^{-3}	0.0
2	2.97×10^{-4}	-5.78×10^{-5}	6.67×10^{-3}
3	0.0	0.0	-1.83×10^{-2}

Table 4: NFORC output at node 2.

Step	NFORC1	NFORC2	NFORC3
1	1.39	0.574	0.0
2	−0.653	-2.31×10^{-2}	−2.00
3	0.0	0.0	2.50

Table 5: Mode shape displacement components at node 10.

Mode	Eigenvalue	u_1	u_2	u_3
1	20.0	0.327	0.624	0.0
2	30.0	0.515	−0.653	0.0
3	40.0	−0.144	1.0	0.0

Table 6: Mode shape rotation components at node 10.

Mode	Eigenvalue	ϕ_1	ϕ_2	ϕ_3
1	20.0	0.0	0.0	−0.416
2	30.0	0.0	0.0	0.436
3	40.0	0.0	0.0	−0.345

Nonlinear behavior

All results correspond to the increment when the rotation is $3\pi/4$.

Table 7: Displacements at node 1.

Step	u_1	u_2	u_3
1	−3.06	0.561	0.0
2	−3.41	-2.22×10^{-4}	−0.706

Step	u_1	u_2	u_3
3	9.30410×10^{-5}	−0.1451	0.353
4	−3.06	0.561	5.51×10^{-5}

Table 8: NFORC output at node 1.

Step	NFORC1	NFORC2	NFORC3
1	−679	−1080	0.0
2	−1090	−47.7	1120
3	−8.46	−1190	−757
4	−623	−1270	4.44×10^{-2}

Table 9: Displacements at node 2.

Step	u_1	u_2	u_3
1	−2.35	2.27	0.0
2	−3.41	-2.22×10^{-4}	−0.706
3	-9.31×10^{-5}	1.56	−0.354
4	−2.35	2.27	6.87×10^{-5}

Table 10: NFORC output at node 2.

Step	NFORC1	NFORC2	NFORC3
1	−2090	−1420	0.0
2	−935	−95.4	1270
3	186	−313	563
4	−1970	−1820	1.83×10^{-2}

Behavior with a colinear arrangement

Table 11: Displacements at node 10.

Step	u_1	u_2	u_3
1	1.59×10^{-3}	-7.69×10^{-3}	0.0
2	0.0	0.0	-2.06×10^{-3}
3	0.0	0.0	2.06×10^{-3}
4	0.0	0.0	0.0

Table 12: Rotations at node 10.

Step	ϕ_1	ϕ_2	ϕ_3
1	0.0	0.0	3.76×10^{-3}
2	8.36×10^{-4}	-8.36×10^{-4}	0.0

Step	ϕ_1	ϕ_2	ϕ_3
3	-8.36×10^{-4}	8.36×10^{-4}	0.0
4	0.0	0.0	0.0

Table 13: Displacements at node 1.

Step	u_1	u_2	u_3
1	3.45×10^{-4}	-1.72×10^{-4}	0.0
2	0.0	0.0	-1.15×10^{-4}
3	0.0	0.0	1.15×10^{-4}
4	0.0	0.0	0.0

Table 14: NFORC output at node 2.

Step	NFORC1	NFORC2	NFORC3
1	0.483	–0.483	0.0
2	0.0	0.0	–0.483
3	0.0	0.0	0.483
4	0.0	0.0	0.0

Table 15: Mode shape displacement components at node 10.

Mode	Eigenvalue	u_1	u_2	u_3
1	20.0	0.327	0.560	0.0
2	30.0	0.494	–0.523	0.0
3	40.0	0.172	-6.03×10^{-2}	0.0

Table 16: Mode shape rotation components at node 10.

Mode	Eigenvalue	ϕ_1	ϕ_2	ϕ_3
1	20.0	0.0	0.0	–0.259
2	30.0	0.0	0.0	0.241
3	40.0	0.0	0.0	0.259

Input files

exdc2lx1.inp	Linear behavior of DCOUP2D elements with **LOAD CASE*.
exdc3lx1.inp	Linear behavior of DCOUP3D elements.
exdc2nx1.inp	Geometrically nonlinear behavior of DCOUP2D elements.
exdc3nx1.inp	Geometrically nonlinear behavior of DCOUP3D elements.
exdc3cx1.inp	Test of DCOUP3D elements with colinear coupling nodes.

Drag chain elements

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

B21 B33H DRAG2D DRAG3D

Problem description

Model: Each system tested contains a drag chain element attached to a beam element, which is fully restrained at the other end. For the two-dimensional case a B21 element has a DRAG2D element attached at the second node. A concentrated force is applied in the *y*-direction at the free end. To test the three-dimensional case, a DRAG3D element is attached to a B33H beam element. In the three-dimensional case the seabed lying in the global *x*–*y* plane is modeled as a rigid surface.

DRAG2D:	
Friction limit	125.0
Horizontal length at slip	0.5
DRAG3D:	
Total length of chain	131.0
Friction coefficient	0.3
Weight of chain per unit length	4.0
Length of chain lying on the seabed	104.0
Height of beam above the seabed	10.0

Results and discussion

The calculated reaction forces are in agreement with the applied loads: the applied force is recovered from the forces in the chain elements and reaction forces at the restrained node of the beam.

Input files

dragchaintest_drag2d.inp	DRAG2D element.
dragchaintest_drag3d.inp	DRAG3D element.

Miscellaneous tests

In this section:

- *Rebar in Abaqus/Standard*
- *Rebar in Abaqus/Explicit*
- *Beam rebar line reinforcement in Abaqus/Standard and Abaqus/Explicit*
- *Convection elements: transport of a temperature pulse*
- *Continuum shells: basic element modes*
- *Transverse shear for shear-flexible shells*
- *Linear dynamic analysis with fluid link*
- *Rigid bodies with temperature DOFs, heat capacitance, and nodal-based thermal loads*
- *Analysis of unbounded acoustic regions*
- *One-dimensional steady-state dynamic solutions for poroelastic acoustic elements*
- *Nonstructural mass verification*
- *Mass adjust verification*
- *Defining a reference mesh*
- *Bending stabilization for full-integration linear elements*

Rebar in Abaqus/Standard

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Rebars in membranes

Elements tested

M3D4 M3D4R M3D8 M3D8R

Problem description

These tests verify the modeling of element reinforcements in membrane elements. The rebar option is tested in the areas of kinematics, prestressing of the rebar, compatibility with material property definitions, and compatibility with prescribed temperatures and field variables. All membranes that allow rebar are tested and compared to continuum and shell elements. Each input file contains tests for membrane, continuum, and shell elements.

Kinematics are tested by applying a uniaxial displacement with various rebar orientations. In the first test rebar are placed along the x-axis, and a displacement is prescribed in the x-direction. In the second test rebar are oriented at $\pm 30°$ from the x-axis. Again, a prescribed displacement is applied along the x-axis. In the third test rebar are oriented along the y-axis, and a displacement is prescribed in the x-direction. The fourth test includes large geometry changes. The rebar are initially defined at $\pm 30°$ from the x-axis. A large displacement is prescribed in the x-direction and causes the orientation of the rebar to change because of the large shearing strains. The fifth and sixth tests define various rebar orientations. In the seventh test rebar angle output is measured with respect to the second isoparametric direction.

The material test includes five combinations of material definitions for the base element and the rebar. For each combination a single element is loaded with a prescribed uniaxial displacement. Elastic, elastic-plastic, hyperelastic, and hypoelastic material properties are used. The combinations are as follows: elastic base and elastic rebar, elastic base and elastic-plastic rebar, elastic-plastic base and elastic rebar, hyperelastic base and elastic rebar, and elastic base and hypoelastic rebar.

Thermal expansion of the rebar is tested by constraining all the degrees of freedom of the elements and applying a temperature load. The rebar is positioned along the x-axis. The base material is dependent on temperature and the first field variable. The rebar properties are dependent on the second field variable. Step 1 uniformly increases the temperature from 0° to 100°, with both field variables set to 1. Step 2 increases the first field variable from 0 to 1, and Step 3 increases the second field variable from 0 to 1.

Initial stresses are tested in two ways. The tests consist of a single underlying membrane element with isoparametric rebar. In the first test an initial tensile stress is applied to the rebar, and no initial stresses are applied to the underlying membrane element. Thus, the membrane element will compress, and the initial rebar tensile stress will be reduced until equilibrium with the underlying solid is reached. The second test applies an initial tensile stress to the rebar but forces this initial stress to remain constant by means of holding prestress in rebar. The stress in the rebar remains unchanged, whereas the underlying membrane deforms to equilibrate the rebar stress.

Input file *em_postoutput.inp* tests the postprocessing output procedure and ensures that rebar output quantities are written properly to the restart file.

Input file *em_nodalthick.inp* tests variable thickness shells and membranes containing rebar. The nodal thickness procedure specifies a linearly varying element thickness.

Results and discussion

The results agree with the analytically obtained values.

Input files

em_kinematics1.inp	Rebar, 0° orientation.
em_kinematics2.inp	Rebar, 30° orientation.
em_kinematics3.inp	Rebar, 90° orientation.
em_kinematics4.inp	Rebar, 30° orientation, finite strains.
em_kinematics5.inp	Rebar, defined using the ORIENTATION parameter on **REBAR LAYER*.
em_kinematics6.inp	Rebar, referencing user-defined **ORIENTATION*.
em_kinematics6.f	User subroutine *ORIENT* used in em_kinematics6.inp.
em_kinematics7.inp	Rebar, test of rebar angle output measured with respect to the second isoparametric direction.
em_material.inp	Rebar, 0° orientation, test of material combinations, perturbation step with **LOAD CASE*.
em_thermal.inp	Rebar, 0° orientation, test of temperature and field variable dependence.
em_prestress.inp	Rebar, 0° orientation, test of initial stresses with and without **PRESTRESS HOLD*.
em_prestress.f	User subroutine *SIGINI* used in em_prestress.inp.
em_postoutput.inp	Rebar, postprocessing with the **POST OUTPUT* option.
em_nodalthick.inp	Rebar, variable thicknesses using the **NODAL THICKNESS* option.

Rebars in surface elements

Elements tested

SFM3D3 SFM3D4 SFM3D4R SFM3D6 SFM3D8 SFM3D8R

Problem description

Model: Similar to the one used for rebars in membranes.

Material: Similar to the one used for rebars in membranes.

Results and discussion

The results agree with those for rebars in membranes when the material stiffness for the membranes is set nearly to zero.

Input files

ex_kinematics1.inp	Rebar, 0° orientation.
ex_kinematics2.inp	Rebar, 30° orientation.
ex_kinematics3.inp	Rebar, 30° orientation, finite strains.

ex_kinematics4.inp	Rebar, defined using the ORIENTATION parameter on **REBAR LAYER*.
ex_kinematics5.inp	Rebar, referencing user-defined **ORIENTATION*.
ex_kinematics5.f	User subroutine *ORIENT* used in ex_kinematics5.inp.
ex_material.inp	Rebar, 0° orientation, test of material combinations.
ex_thermal.inp	Rebar, 0° orientation, test of temperature and field variable dependence.
ex_prestress.inp	Rebar, 0° orientation, test of initial stresses with and without **PRESTRESS HOLD*.
ex_prestress.f	User subroutine *SIGINI* used in ex_prestress.inp.

Rebars in general shells

Elements tested

S4 S4R S8R S8R5 SC8R

Problem description

Model:

Planar dimensions	10×10
Thickness	2.0 (for tensile test), 10.0 (for bending test)

Material:

Young's modulus of bulk material	1.0 (for tensile test), 3×10^6 (for bending test)
Young's modulus of rebar	30×10^6
Poisson's ratio of both materials	0.0
Reinforcement for tensile test	REBAR1, 1., 2.5, 0., RBMAT, 0, 1
	REBAR2, 1., 2.5, 0., RBMAT, 90, 1
	REBAR3, 1., 3.5355, 0., RBMAT, 45, 1
	REBAR4, 1., 3.5355, 0., RBMAT, 135, 1
Reinforcement for bending test	REBAR, .1, 2.5, −2.5, RBMAT, 0, 1

Results and discussion

The results agree with the analytically obtained values.

Input files

ese4sxr4.inp	S4 elements; tension with rebar; 0° orientation, 45° orientation, 90° orientation, and 135° orientation.
ese4sxr3.inp	S4 elements; bending with rebar; 0° orientation.
esf4sxr4.inp	S4R elements; tension with rebar; 0° orientation, 45° orientation, 90° orientation, and 135° orientation.
esf4sxr3.inp	S4R elements; bending with rebar; 0° orientation.
es68sxr4.inp	S8R elements; tension with rebar; 0° orientation, 45° orientation, 90° orientation, and 135° orientation.

es68sxr3.inp	S8R elements; bending with rebar; 0° orientation.
es58sxrd.inp	S8R5 elements; bending with rebar; 0° orientation; response spectrum.
esc8sxr4.inp	SC8R elements; tension with rebar; 0° orientation, 45° orientation, 90° orientation, and 135° orientation.
esc8sxr3.inp	SC8R elements; bending with rebar; 0° orientation.

Rebars in axisymmetric membranes

Elements tested

MAX1 MAX2 MGAX1 MGAX2

Problem description

Model:

Length	5.0
Midsurface radius	2.0
Thickness	0.05

Material:

Young's modulus of bulk material	1.0×10^5
Young's modulus of rebar	1.0×10^8
Poisson's ratio of both materials	0.495
Reinforcement for tension and torsion tests	REBAR, 0.005, 0.31416, 0, RBMAT, 50

Results and discussion

If rebars are not axial (rebar angle 0°) or circumferential (rebar angle 90°), element types MGAX1 and MGAX2 predict twist under axial tension (Step 1 in all the input files). The twist angle is determined by the initial rebar angle and the material properties. If the Poisson's ratio of the material is sufficiently different from zero, the twist angle changes sign at some intermediate rebar angle between 0° and 90°. This result is accompanied by a change in sign of the stress in the rebar. This behavior is illustrated in *Figure 1*(a), where results for the twist angle are shown for element types MGAX1, MGAX2, and CGAX4R (axisymmetric continuum element with twist) when both the rebar and the bulk materials are almost incompressible. *Figure 1*(b) shows the evolution of this behavior with the Poisson's ratios of the materials. For $\boldsymbol{\nu} =$ 0.05 the twist angle does not change sign as the initial rebar angle changes from 0° to 90°.

Input files

ema2srri.inp	MAX1 elements, tension.
ema3srri.inp	MAX2 elements, tension.
emg2srri.inp	MGAX1 elements, tension and torsion
emg3srri.inp	MGAX2 elements, tension and torsion.

Rebars in axisymmetric surface elements

Elements tested

SFMAX1 SFMAX2 SFMGAX1 SFMGAX2

Problem description

Model: Similar to the one used for rebars in axisymmetric membranes.

Material: Similar to the one used for rebars in axisymmetric membranes.

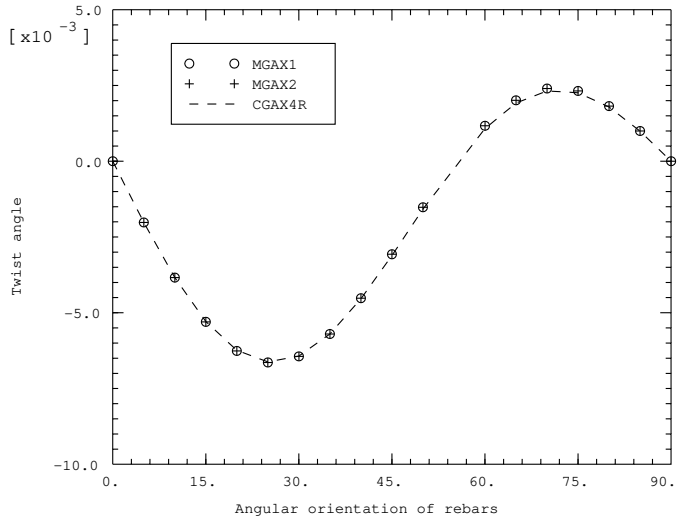
Results and discussion

The results agree with those for rebars in axisymmetric membranes when the material stiffness for the membranes is set nearly to zero.

Input files

exa2srri.inp	SFMAX1 elements, tension.
exa3srri.inp	SFMAX2 elements, tension.
exg2srri.inp	SFMGAX1 elements, tension and torsion
exg3srri.inp	SFMGAX2 elements, tension and torsion.

Figures



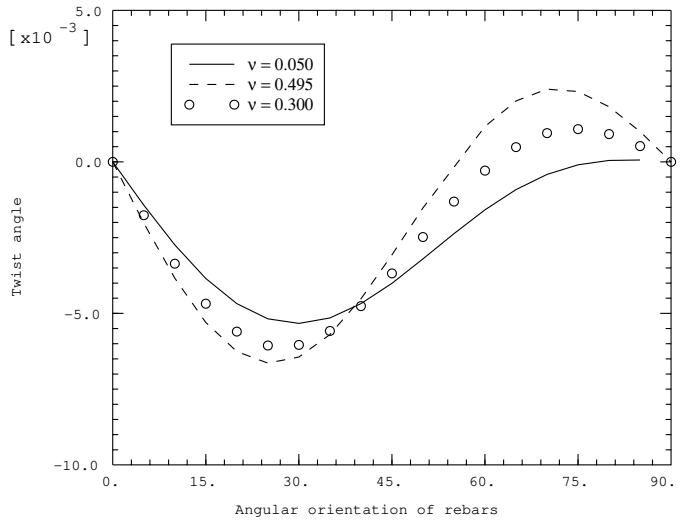


Figure 1: Variation of twist with rebar angle.

Rebars in axisymmetric shells

Elements tested

SAX1 SAX2

Problem description

Model:

Length	10.0
Inside radius for hoop test	5.0 (Flat solid disk for radial test)

Thickness	2.0

Material:

Young's modulus of bulk material	1.0
Young's modulus of rebar	30×10^6
Poisson's ratio of both materials	0.0
Reinforcement for hoop test	REBAR1, 1, 2.5, –1, RBMAT, 90
	REBAR2, 1, 2.5, 1, RBMAT, 90
Reinforcement for radial test	REBAR, 1, 46.245, 0, RBMAT, 0

Results and discussion

The results agree with the analytically obtained values.

Input files

esa2sxrh.inp	SAX1 elements, hoop rebar.
esa2sxrr.inp	SAX1 elements, radial rebar using the GEOMETRY=ANGULAR parameter on **REBAR LAYER*.
esa3sxrh.inp	SAX2 elements, hoop rebar.
esa3sxrr.inp	SAX2 elements, radial rebar using the GEOMETRY=ANGULAR parameter on **REBAR LAYER*.

Rebars in general surface elements embedded in three-dimensional solids

Elements tested

C3D8 C3D20 SFM3D4R SFM3D8R

Problem description

Model:

Cubic dimension	$10.0 \times 10.0 \times 10.0$

Material:

Young's modulus of bulk material	1.0
Young's modulus of rebar	30×10^6
Poisson's ratio of both materials	0.0
Reinforcement	REBAR, 1., 2.5, 0., RBMAT, 0, 1

Results and discussion

The results agree with the analytically obtained values.

Input files

ec38sfrg.inp	C3D8 with SFM3D4R elements, rebar with 0° orientation.
ec3ksfrg.inp	C3D20 with SFM3D8R elements, rebar with 0° orientation.

Rebars in axisymmetric surface elements embedded in axisymmetric solids and axisymmetric solids with twist

Elements tested

CAX4 CAX8 CGAX4 CGAX4R CGAX4T CGAX8 CGAX8T SFMAX1 SFMAX2 SFMGAX1 SFMGAX2

Problem description

Model:

Planar dimensions	10.0×10.0
Inside radius	0.0

Material:

Young's modulus of bulk material	1.0
Young's modulus of rebar	30×10^6
Poisson's ratio of both materials	0.0
Reinforcement for hoop test	REBAR1, .04, .3333, 0., RBMAT, 90
Reinforcement for radial test	REBAR2, .04, 46.245, 0., RBMAT, 0

Results and discussion

The results agree with the analytically obtained values.

Input files

eca4sfri.inp	CAX4 elements with SFMAX1 elements, hoop rebar, and radial rebar using the GEOMETRY=ANGULAR parameter on **REBAR LAYER*.
eca4sfr2.inp	CAX4 elements with SFMAX1 elements, radial rebar using the GEOMETRY=ANGULAR parameter on **REBAR LAYER*.
eca4sfrs.inp	CAX4 elements with SFMAX1 elements, hoop rebar, and radial rebar using the GEOMETRY=ANGULAR parameter on **REBAR LAYER*.
eca8sfri.inp	CAX8 elements with SFMAX2 elements, hoop rebar, and radial rebar using the GEOMETRY=ANGULAR parameter on **REBAR LAYER*.
eca8sfr2.inp	CAX8 elements with SFMAX2 elements, radial rebar using the GEOMETRY=ANGULAR parameter on **REBAR LAYER*.
eca8sfrs.inp	CAX8 elements with SFMAX2 elements, hoop rebar, and radial rebar using the GEOMETRY=ANGULAR parameter on **REBAR LAYER*.
eca4gfri.inp	CGAX4 elements with SFMGAX1 elements, hoop rebar, and radial rebar using the GEOMETRY parameter on **REBAR LAYER*.

eca4gfrs.inp	CGAX4 elements with SFMGAX1 elements, hoop rebar, and radial rebar using the GEOMETRY=ANGULAR parameter on **REBAR LAYER*.
eca4gfr2.inp	CGAX4 elements with SFMGAX1 elements, radial rebar using the GEOMETRY parameter on **REBAR LAYER*.
eca4hfri.inp	CGAX4T elements with SFMGAX1 elements, hoop rebar, and radial rebar using the GEOMETRY=ANGULAR parameter on **REBAR LAYER*.
eca4hfrs.inp	CGAX4T elements with SFMGAX1 elements, hoop rebar, and radial rebar using the GEOMETRY parameter on **REBAR LAYER*.
eca4hfr2.inp	CGAX4T elements with SFMGAX1 elements, radial rebar using the GEOMETRY parameter on **REBAR LAYER*.
eca8gfri.inp	CGAX8 elements with SFMGAX2 elements, hoop rebar, and radial rebar using the GEOMETRY=ANGULAR parameter on **REBAR LAYER*.
eca8gfrs.inp	CGAX8 elements with SFMGAX2 elements, hoop rebar, and radial rebar using the GEOMETRY=ANGULAR parameter on **REBAR LAYER*.
eca8gfr2.inp	CGAX8 elements with SFMGAX2 elements; radial rebar using the GEOMETRY parameter on **REBAR LAYER*.
eca8hfri.inp	CGAX8T elements with SFMGAX2 elements, hoop rebar, and radial rebar using the GEOMETRY=ANGULAR parameter on **REBAR LAYER*.
eca8hfrs.inp	CGAX8T elements with SFMGAX2 elements, hoop rebar, and radial rebar using the GEOMETRY=ANGULAR parameter on **REBAR LAYER*
eca8hfr2.inp	CGAX8T elements with SFMGAX2 elements; radial rebar using the GEOMETRY=ANGULAR parameter on **REBAR LAYER*.

Rebars in plane stress and plane strain solids

Elements tested

CPE4 CPE8 CPS4 CPS8

Problem description

Model:

Planar dimension	10.0×10.0
Thickness	1.0

Material:

Young's modulus of bulk material	1.0
Young's modulus of rebar	30×10^6
Reinforcement	
Isoparametric:	Skew:
PLANE, .04, .25, 0., .25, 2	PLANE, .04, .25, 0.
PLANE, .04, .25, 0., .50, 2	.5, .5
PLANE, .04, .25, 0., .75, 2	PLANE, .04, .25, 0.
PLANE, .04, .25, 0., .25, 1	0., 1., 0., 1.
PLANE, .04, .25, 0., .50, 1	PLANE, .04, .25, 0.
PLANE, .04, .25, 0., .75, 1	0., 0., .5, .5

PLANE, .04, .25
0., .5, 0., 0., .5
PLANE, .04, .25, 0.
1., 0., 1.
PLANE, .04, .25, 0.
0., .5, .5

Results and discussion

The results agree with the analytically obtained values.

Input files

ece4sfrg.inp	CPE4 elements, isoparametric and skew rebar.
ecs4sfrg.inp	CPS4 elements, isoparametric and skew rebar.
ece8sfrg.inp	CPE8 elements, isoparametric and skew rebar.
ecs8sfrg.inp	CPS8 elements, isoparametric and skew rebar.
ecs4sfrd.inp	CPS4 elements, isoparametric and skew rebar, linear dynamic (**FREQUENCY*, **STEADY STATE DYNAMICS*).

Single rebars in three-dimensional solids

Elements tested

C3D8 C3D8S C3D8HS C3D20

Problem description

Model:

Cubic dimension	$10.0 \times 10.0 \times 10.0$

Material:

Young's modulus of bulk material	1.0
Young's modulus of rebar	30×10^6
Poisson's ratio of both materials	0.0
Reinforcement for single rebar test	BRICK, 1., .5, .5, 1
	BRICK, 1., .5, .5, 2
	BRICK, 1., .5, .5, 3

Results and discussion

The results agree with the analytically obtained values.

Input files

ec38sfr1.inp	C3D8 elements, single rebar.
c3d8s_rebar.inp	C3D8S elements, single rebar.
c3d8hs_rebar.inp	C3D8HS elements, single rebar.
ec3ksfr1.inp	C3D20 elements, single rebar

Single rebar in axisymmetric solids and axisymmetric solids with twist

Elements tested

CAX4 CAX8 CGAX4 CGAX4R CGAX4T CGAX8 CGAX8T

Problem description

Model:

Planar dimensions	10.0×10.0
Inside radius	0.0

Material:

Young's modulus of bulk material	1.0
Young's modulus of rebar	30×10^6
Poisson's ratio of both materials	0.0
Reinforcement for single hoop rebar test	
AXSOL, .4, .25, .25	
AXSOL, .4, .50, .25	
AXSOL, .4, .75, .25	
AXSOL, .4, .25, .50	
AXSOL, .4, .50, .50	
AXSOL, .4, .75, .50	
AXSOL, .4, .25, .75	
AXSOL, .4, .50, .75	
AXSOL, .4, .75, .75	

Results and discussion

The results agree with the analytically obtained values.

Input files

eca4sfr2.inp	CAX4 elements, single hoop rebar.
eca8sfr2.inp	CAX8 elements, single hoop rebar.
eca4gfrn.inp	CGAX4 elements, single hoop rebar.
eca4gfr2.inp	CGAX4 elements, single hoop rebar.
eca4hfrn.inp	CGAX4T elements, single hoop rebar.

eca4hfr2.inp	CGAX4T elements, single hoop rebar.
eca8gfr2.inp	CGAX8 elements, single hoop rebar.
eca8hfr2.inp	CGAX8T elements, single hoop rebar.

Rebars in beams

Elements tested

B23

Problem description

Model:

Length	10.0 (300.0 in file *eb2arxrd.inp*)
Cross-section	10.0 × 10.0 rectangular

Material:

Young's modulus of bulk material	1.0 (for tensile test), 3×10^6 (for bending test)
Young's modulus of rebar	30×10^6
Reinforcement for tensile test	BEAM, 1., −2.5, −2.5
	BEAM, 1., 2.5, 2.5
Reinforcement for bending test	BEAM, 1., −2.5, −2.5
	BEAM, 1., 2.5, −2.5

Results and discussion

The results agree with the analytically obtained values.

Input files

eb2arxrt.inp	B23 elements, tension.
eb2arxrb.inp	B23 elements, bending.
eb2arxrd.inp	B23 elements, bending, linear dynamic (**FREQUENCY*, **MODAL DYNAMIC*).

Rebars with geometry defined by angular spacing and lift equation

Elements tested

SAX2 MAX2 SFMAX2 S4R M3D4R SFM3D4R

Problem description

These tests verify reinforcement with spacing that varies as a function of radial position and reinforcement defined by the tire lift equation. Each input file contains two models; one model contains reinforcement with angular spacing and the other model contains reinforcement defined with the lift equation. Aside from the reinforcement geometry, the two models are identical, consisting of an axisymmetric disk with internal radius of 2.0, external radius of 5.0 and thickness of 0.1. The interior edges of the disks are fully constrained and a prescribed displacement of 1.0×10^{-4} is applied to the exterior edges.

One layer of rebar is defined in the model containing rebar with angular spacing. The rebar is oriented along the radial direction. The second model contains 8 layers of rebar, oriented at an angle of 45°, 135°, 225°, 315°, −45°, −135°, −225°, −315° respectively in the uncured configuration.

Material:

Young's modulus of bulk material	1.0×10^3
Young's modulus of rebar	1.0×10^8
Poisson's ratio of both materials	0.3

Results and discussion

The results agree with the analytically obtained values.

Input files

exa2srrr.inp	SFMAX2 elements.
ex34srrr.inp	SFM3D4R elements. Model is generated by revolving the axisymmetric cross-section defined in exa2srrr.inp
ex34srrl.inp	SFM3D4R elements. Model is generated by reflecting the model defined in ex34srrr.inp
ema2srrr.inp	MAX2 elements.
em34srrr.inp	M3D4R elements. Model is generated by revolving the axisymmetric cross-section defined in ema2srrr.inp
em34srp0.inp	M3D4R elements. Reference model for import.
em34srpx.inp	M3D4R elements. Import from standard to explicit. Requires restart file generated from em34srp0.inp
em34srps.inp	M3D4R elements. Import from explicit to standard. Requires restart file generated from em34srpx.inp
esa2srrr.inp	SAX2 elements.
es34srrr.inp	S4R elements. Model is generated by revolving the axisymmetric cross-section defined in esa2srrr.inp

Rebar in Abaqus/Explicit

Products: Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

CPS4R CPE4R CAX4R C3D8 C3D8R M3D3 M3D4R SFM3D4R SAX1 S3R S3RS S4 S4R
SC8R S4RS S4RSW

Problem description

This test verifies the modeling of element reinforcements with the element-based rebar procedure. The layers of reinforcement are defined as part of the element section definition. These options are tested in the areas of kinematics, compatibility with material property definitions, and compatibility with prescribed temperatures and field variables. All element types that support reinforcement are tested. The procedure to specify layers of reinforcement defined as part of the element section definition is used for shell, surface, and membrane elements. The element-based rebar procedure is used for continuum elements.

Kinematics of rebar in continuum elements

Continuum element kinematics are tested in two ways. In the first test rebar are placed at various locations and orientations within an element and a uniaxial displacement is applied to the element. The rebar are located one-third of the distance from the element edge and are given orientation angles of 0, 45, and 90°. For plane strain and plane stress elements 89.9° is used instead of 90° since a rebar oriented at 90° for these elements would provide no stiffness. Rebar are also placed directly along the element edges with orientation angles of 0°. The second test checks that the rebar yield the correct strains for various deformation modes. Rebar are positioned at one-third of the distance from the lower edge in a CPE4R element. Uniaxial stretching is performed in the direction of the rebar and in the direction perpendicular to the rebar. Simple shear is tested with the rebar parallel to the direction of motion and with the rebar perpendicular to the direction of motion.

Kinematics of rebar in shell elements

Three tests exist for rebar in shells. The first two tests cover kinematics of rebar placed at the midsurface in shells. The third test covers bending behavior of shells in which rebar are placed away from the midsurface.

The first kinematics test, rebar_*elementtype*.inp, places the rebar at various orientations within an element, and a uniaxial displacement is applied to the element. The rebars are defined at orientation angles of 0, 30, and 90°. This test is repeated for elements in which the nodal thicknesses are defined and for composite shells.

The second kinematics test, *rebar_modes.inp*, verifies that the rebar yield the correct strains for various deformation modes. Uniaxial stretching is performed in the direction of the rebar and in the direction perpendicular to the rebar. For general shell and membrane elements simple shear is tested with the rebar parallel to the direction of motion and with the rebar perpendicular to the direction of motion. See *Figure 1*.

The third kinematics test, *rebar_bending.inp*, verifies the bending behavior of shell elements that have undergone finite membrane strains. A finite uniaxial stretch is prescribed at the midsurface of the shell, followed by a rotation at one end of the shell element. This test is repeated for shell elements in which the nodal thicknesses

are defined, with shell elements in which the midsurface position is defined by an offset, and with composite shell elements.

Kinematics of rebar in membrane and surface elements

Two tests exist for rebar in membrane and surface elements. The two tests cover kinematics of rebar placed at the midsurface in membranes and in surface elements and are similar to the first two tests for the shell elements.

Rebar material tests

The material test includes five combinations of material definitions for the base element and for the rebar. For each combination CPE4R, M3D4R, S4R, and S4RS elements are loaded with a prescribed uniaxial displacement. Elastic, elastic-plastic, and hyperelastic material properties are used for both the base element and the rebar. The combinations are as follows: elastic base and elastic rebar, elastic base and elastic-plastic rebar, elastic-plastic base and elastic rebar, hyperelastic base and elastic rebar, and hyperelastic base and hyperelastic rebar.

Thermal expansion of rebar in continuum elements

Thermal expansion of the rebar is tested by constraining all degrees of freedom of the elements and applying a temperature load. The rebar is positioned one-third of the distance from the element's lower edge. The temperature on the lower edge is increased from 0 to 20°, while the temperature on the top edge is increased from 0 to 80°.

Thermal expansion of rebar in shell and membrane elements

Thermal expansion of the rebar is tested by constraining all degrees of freedom of the elements and applying a temperature load. The rebar is placed at the midsurface in membranes and at one-third of the thickness from the bottom surface in shells.

The nodal temperatures of membrane elements are increased uniformly from 0 to 40°. The nodal temperatures of shell elements are increased uniformly throughout the element but vary through the thickness of the shell. The temperatures are applied in two ways: as a midsurface temperature that is increased from 0 to 50° along

with a temperature gradient through the shell thickness that is increased from 0 to $30°/\boldsymbol{L}$, and directly at the section points through the shell thickness.

Temperature- and field-variable-dependent rebar materials

The use of temperature- and field-variable-dependent inelastic material properties is tested by stretching the rebar until yield occurs, while simultaneously applying a uniform temperature or field variable increase. The underlying elements are modeled with an elastic material.

Body loads on elements containing rebar

This test applies a body force and a gravity load to all elements that allow rebar. All degrees of freedom are fixed, and the reaction forces are output. Gravity loads are based on the magnitude of the user-provided gravity constant, the element density, and element volume; the body forces are based on the body force magnitude and the element volume. Since the mass of the rebar is considered significant and is added to the total mass of the element, the rebar will contribute to the gravity load. The volume of the rebar, however, is not added to the total element volume since the rebars are considered to be embedded in the underlying element. Therefore, rebar will not contribute to body forces.

Prestress in elements containing rebar

This test consists of shell, membrane and continuum elements with isoparametric rebar. An initial tensile stress is applied to the rebar, and no initial stresses are applied to the underlying elements. Thus, the underlying elements will compress, and the initial rebar tensile stress will be reduced until equilibrium between the two is reached.

Results and discussion

The results for all the test cases agree with the analytical values that have been included at the top of each input file.

Input files

Input files that use the element-based rebar procedure

rebar_cpe4r.inp	Kinematics test for the CPE4R element.
rebar_cax4r.inp	Kinematics test for the CAX4R element.
rebar_cps4r.inp	Kinematics test for the CPS4R element.
rebar_c3d8.inp	Kinematics test for the C3D8 element.
rebar_c3d8r.inp	Kinematics test for the C3D8R element.

Input files that use the procedure to specify layers of reinforcement defined as part of the element section definition

rebar_m3d4r.inp	Kinematics test for the M3D4R element.
rebar_sfm3d4r.inp	Kinematics test for the SFM3D4R element.
rebar_sax1.inp	Kinematics test for the SAX1 element.
rebar_s4.inp	Kinematics test for the S4 element.
rebar_s4r.inp	Kinematics test for the S4R element.
rebar_sc8r.inp	Kinematics test for the SC8R element.
rebar_s4rs.inp	Kinematics test for the S4RS element.
rebar_s4rsw.inp	Kinematics test for the S4RSW element.
rebar_orient.inp	Rebar orientation test for shells and membranes.
rebar_bending.inp	Shell rebar bending test.

Input files that use the procedures to specify layers of reinforcement defined as part of the element section definition as well as element-based rebar

rebar_modes.inp	Multiple deformation modes.
rebar_material.inp	Rebar material test.
rebar_prestress.inp	Test of initial rebar stresses.
rebar_tempdep.inp	Temperature-dependent rebar material test.
rebar_fielddep.inp	Field-variable-dependent rebar material test.
rebar_thermalexp.inp	Rebar thermal expansion test.
rebar_bodyload.inp	Body and gravity load test of rebar.

Figures

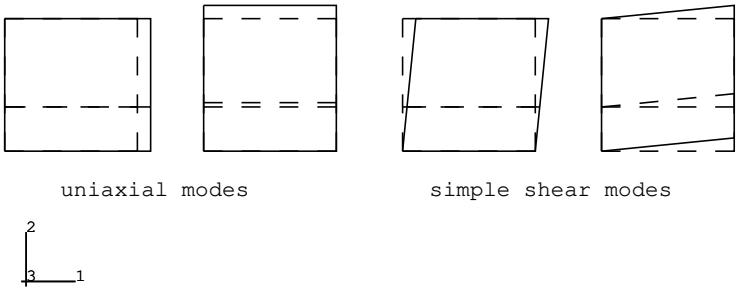


Figure 1: Deformation modes for rebar in a CPE4R, M3D4R, and S4R element.

Beam rebar line reinforcement in Abaqus/Standard and Abaqus/Explicit

Products: Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

B21 B22 B31 B32 PIPE21 PIPE31

Problem description

These tests verify the modeling of beam element reinforcements in both Abaqus/Standard and Abaqus/Explicit. The rebar line reinforcement is defined as part of the element section definition. These options are tested in kinematics, multistep, restart, import, prestress, and mass scaling analyses. All beam element types that support rebar reinforcement are tested. The procedure to specify rebar reinforcement defined as part of the element section definition is used in the tests.

Kinematics of rebar in beam elements

Kinematics for beam elements with rebar reinforcement are tested under uniaxial stretch, twist, and pure bending deformation modes. Rebar lines are placed at various locations within the beam elements. All supported beam element types including B21, B22, B31, B32, PIPE21, and PIPE31 are tested. In addition, for three-dimensional beam element types B31 and B32, different section types including ARBITRARY, BOX, CIRC, HEX, I, L, RECT, and TRAPEZOID are tested. These tests check that the rebar exhibit the correct stresses and strains for the three deformation modes. For each test case the results from Abaqus/Explicit agree with the results from Abaqus/Standard.

In addition, a beam structure with a uniformly distributed load is tested in both Abaqus/Standard and Abaqus/Explicit. The beam structure contains 20 B31 elements with embedded rebar line reinforcement. The run time for the Abaqus/Explicit test is set long enough to make sure it reaches the equilibrium state, and the results agree with those from the Abaqus/Standard analysis (see *Figure 1*) .

Multistep and restart tests of beam elements with rebar line reinforcement

These tests cover kinematics of rebars in beam elements under multistep and restart analyses. All supported beam element types including B21, B22, B31, B32, PIPE21, and PIPE31 are tested under prescribed uniaxial stretch, twist, and pure bending conditions. Each test case is tested in Abaqus/Standard and Abaqus/Explicit, and the results of the tests from Abaqus/Explicit agree with those from Abaqus/Standard.

Import tests of beam elements with rebar line reinforcement

These tests include Abaqus/Standard to Abaqus/Explicit, Abaqus/Explicit to Abaqus/Standard, and Abaqus/Explicit to Abaqus/Explicit import analyses for the beam elements with rebar line reinforcement. All supported beam element types including B21, B22, B31, B32, PIPE21, and PIPE31 are tested under prescribed uniaxial stretch, twist, and pure bending conditions. States including stresses, strains, and rebar forces on import agree between the first analyses and the import analyses.

Prestress in beam elements with rebar line reinforcement

Initial stresses are applied to the rebar and to the underlying beam elements. Different beam element types including B21, B22, B31, and B32 are tested; and the initial stresses on the rebar and beam elements agree with the user-specified values.

Mass scaling for beam elements with rebar line reinforcement

These problems test mass scaling for beam elements with rebar line reinforcement. Both fixed mass scaling and variable mass scaling are tested for different beam types including B21, B22, B31, and B32. Stable time increments show the expected scaled values, and the stress and strain results agree with the cases without mass scaling.

Results and discussion

The results for all the test cases in Abaqus/Explicit agree with the results in Abaqus/Standard.

Input files

Input files for kinematics tests of beam elements with rebar reinforcement

b21_rebar_std.inp	Abaqus/Standard kinematics test for B21 element.
b21_rebar_xpl.inp	Abaqus/Explicit kinematics test for B21 element.
b22_rebar_std.inp	Abaqus/Standard kinematics test for B22 element.
b22_rebar_xpl.inp	Abaqus/Explicit kinematics test for B22 element.
b31_arb_c_rebar_std.inp	Abaqus/Standard kinematics test for B31 element with closed ARBITRARY section type.
b31_arb_c_rebar_xpl.inp	Abaqus/Explicit kinematics test for B31 element with closed ARBITRARY section type.
b31_arb_o_rebar_std.inp	Abaqus/Standard kinematics test for B31 element with open ARBITRARY section type.
b31_arb_o_rebar_xpl.inp	Abaqus/Explicit kinematics test for B31 element with open ARBITRARY section type.
b31_box_rebar_std.inp	Abaqus/Standard kinematics test for B31 element with BOX section type.
b31_box_rebar_xpl.inp	Abaqus/Explicit kinematics test for B31 element with BOX section type.
b31_circ_rebar_std.inp	Abaqus/Standard kinematics test for B31 element with CIRC section type.
b31_circ_rebar_xpl.inp	Abaqus/Explicit kinematics test for B31 element with CIRC section type.
b31_hex_rebar_std.inp	Abaqus/Standard kinematics test for B31 element with HEX section type.
b31_hex_rebar_xpl.inp	Abaqus/Explicit kinematics test for B31 element with HEX section type.
b31_i_rebar_std.inp	Abaqus/Standard kinematics test for B31 element with I section type.
b31_i_rebar_xpl.inp	Abaqus/Explicit kinematics test for B31 element with I section type.
b31_rect_rebar_std.inp	Abaqus/Standard kinematics test for B31 element with RECT section type.

b31_rect_rebar_xpl.inp	Abaqus/Explicit kinematics test for B31 element with RECT section type.
b31_trap_rebar_std.inp	Abaqus/Standard kinematics test for B31 element with TRAPEZOID section type.
b31_trap_rebar_xpl.inp	Abaqus/Explicit kinematics test for B31 element with TRAPEZOID section type.
b32_arb_c_rebar_std.inp	Abaqus/Standard kinematics test for B32 element with closed ARBITRARY section type.
b32_arb_c_rebar_xpl.inp	Abaqus/Explicit kinematics test for B32 element with closed ARBITRARY section type.
b32_arb_o_rebar_std.inp	Abaqus/Standard kinematics test for B32 element with open ARBITRARY section type.
b32_arb_o_rebar_xpl.inp	Abaqus/Explicit kinematics test for B32 element with open ARBITRARY section type.
b32_box_rebar_std.inp	Abaqus/Standard kinematics test for B32 element with BOX section type.
b32_box_rebar_xpl.inp	Abaqus/Explicit kinematics test for B32 element with BOX section type.
b32_circ_rebar_std.inp	Abaqus/Standard kinematics test for B32 element with CIRC section type.
b32_circ_rebar_xpl.inp	Abaqus/Explicit kinematics test for B32 element with CIRC section type.
b32_hex_rebar_std.inp	Abaqus/Standard kinematics test for B32 element with HEX section type.
b32_hex_rebar_xpl.inp	Abaqus/Explicit kinematics test for B32 element with HEX section type.
b32_i_rebar_std.inp	Abaqus/Standard kinematics test for B32 element with I section type.
b32_i_rebar_xpl.inp	Abaqus/Explicit kinematics test for B32 element with I section type.
b32_rect_rebar_std.inp	Abaqus/Standard kinematics test for B32 element with RECT section type.
b32_rect_rebar_xpl.inp	Abaqus/Explicit kinematics test for B32 element with RECT section type.
b32_trap_rebar_std.inp	Abaqus/Standard kinematics test for B32 element with TRAPEZOID section type.
b32_trap_rebar_xpl.inp	Abaqus/Explicit kinematics test for B32 element with TRAPEZOID section type.
p21_rebar_std.inp	Abaqus/Standard kinematics test for PIPE21 element.
p21_rebar_xpl.inp	Abaqus/Explicit kinematics test for PIPE21 element.
p31_rebar_std.inp	Abaqus/Standard kinematics test for PIPE31 element.
p31_rebar_xpl.inp	Abaqus/Explicit kinematics test for PIPE31 element.
bm-cload-std-rebar.inp	Abaqus/Standard test of beam structure with 20 B31 elements under uniformly distributed load.
bm-cload-xpl-rebar.inp	Abaqus/Explicit test of beam structure with 20 B31 elements under uniformly distributed load.

Input files for multistep and restart tests of beam elements with rebar line reinforcement

b21_rebar_std_2step.inp	Abaqus/Standard multistep test for B21 element.

b21_rebar_std_1.inp	First test of Abaqus/Standard restart analysis for B21 element.
b21_rebar_std_restart.inp	Abaqus/Standard restart test for B21 element.
b21_rebar_xpl_2step.inp	Abaqus/Explicit multistep test for B21 element.
b21_rebar_xpl_1.inp	First test of Abaqus/Explicit restart analysis for B21 element.
b21_rebar_xpl_restart.inp	Abaqus/Explicit restart test for B21 element.
b22_rebar_std_2step.inp	Abaqus/Standard multistep test for B22 element.
b22_rebar_std_1.inp	First test of Abaqus/Standard restart analysis for B22 element.
b22_rebar_std_restart.inp	Abaqus/Standard restart test for B22 element.
b22_rebar_xpl_2step.inp	Abaqus/Explicit multistep test for B22 element.
b22_rebar_xpl_1.inp	First test of Abaqus/Explicit restart analysis for B22 element.
b22_rebar_xpl_restart.inp	Abaqus/Explicit restart test for B22 element.
b31_rebar_std_2step.inp	Abaqus/Standard multistep test for B31 element.
b31_rebar_std_1.inp	First test of Abaqus/Standard restart analysis for B31 element.
b31_rebar_std_restart.inp	Abaqus/Standard restart test for B31 element.
b31_rebar_xpl_2step.inp	Abaqus/Explicit multistep test for B31 element.
b31_rebar_xpl_1.inp	First test of Abaqus/Explicit restart analysis for B31 element.
b31_rebar_xpl_restart.inp	Abaqus/Explicit restart test for B31 element.
b32_rebar_std_2step.inp	Abaqus/Standard multistep test for B32 element.
b32_rebar_std_1.inp	First test of Abaqus/Standard restart analysis for B32 element.
b32_rebar_std_restart.inp	Abaqus/Standard restart test for B32 element.
b32_rebar_xpl_2step.inp	Abaqus/Explicit multistep test for B32 element.
b32_rebar_xpl_1.inp	First test of Abaqus/Explicit restart analysis for B32 element.
b32_rebar_xpl_restart.inp	Abaqus/Explicit restart test for B32 element.
p21_rebar_std_2step.inp	Abaqus/Standard multistep test for PIPE21 element.
p21_rebar_std_1.inp	First test of Abaqus/Standard restart analysis for PIPE21 element.
p21_rebar_std_restart.inp	Abaqus/Standard restart test for PIPE21 element.
p21_rebar_xpl_2step.inp	Abaqus/Explicit multistep test for PIPE21 element.
p21_rebar_xpl_1.inp	First test of Abaqus/Explicit restart analysis for PIPE21 element.
p21_rebar_xpl_restart.inp	Abaqus/Explicit restart test for PIPE21 element.
p31_rebar_std_2step.inp	Abaqus/Standard multistep test for PIPE31 element.
p31_rebar_std_1.inp	First test of Abaqus/Standard restart analysis for PIPE31 element.
p31_rebar_std_restart.inp	Abaqus/Standard restart test for PIPE31 element.
p31_rebar_xpl_2step.inp	Abaqus/Explicit multistep test for PIPE31 element.
p31_rebar_xpl_1.inp	First test of Abaqus/Explicit restart analysis for PIPE31 element.
p31_rebar_xpl_restart.inp	Abaqus/Explicit restart test for PIPE31 element.

Input files for import tests of beam elements with rebar line reinforcement

b21_sx_s_rebar.inp	First test of Abaqus/Standard to Abaqus/Explicit import analysis for B21 element.
b21_sx_x_rebar.inp	Second test of Abaqus/Standard to Abaqus/Explicit import analysis for B21 element.
b21_xs_x_rebar.inp	First test of Abaqus/Explicit to Abaqus/Standard import analysis for B21 element.

b21_xs_s_rebar.inp	Second test of Abaqus/Explicit to Abaqus/Standard import analysis for B21 element.
b21_xx_x1_rebar.inp	First test of Abaqus/Explicit to Abaqus/Explicit import analysis for B21 element.
b21_xx_x2_rebar.inp	Second test of Abaqus/Explicit to Abaqus/Explicit import analysis for B21 element.
b22_sx_s_rebar.inp	First test of Abaqus/Standard to Abaqus/Explicit import analysis for B22 element.
b22_sx_x_rebar.inp	Second test of Abaqus/Standard to Abaqus/Explicit import analysis for B22 element.
b22_xs_x_rebar.inp	First test of Abaqus/Explicit to Abaqus/Standard import analysis for B22 element.
b22_xs_s_rebar.inp	Second test of Abaqus/Explicit to Abaqus/Standard import analysis for B22 element.
b22_xx_x1_rebar.inp	First test of Abaqus/Explicit to Abaqus/Explicit import analysis for B22 element.
b22_xx_x2_rebar.inp	Second test of Abaqus/Explicit to Abaqus/Explicit import analysis for B22 element.
b31_sx_s_rebar.inp	First test of Abaqus/Standard to Abaqus/Explicit import analysis for B31 element.
b31_sx_x_rebar.inp	Second test of Abaqus/Standard to Abaqus/Explicit import analysis for B31 element.
b31_xs_x_rebar.inp	First test of Abaqus/Explicit to Abaqus/Standard import analysis for B31 element.
b31_xs_s_rebar.inp	Second test of Abaqus/Explicit to Abaqus/Standard import analysis for B31 element.
b31_xx_x1_rebar.inp	First test of Abaqus/Explicit to Abaqus/Explicit import analysis for B31 element.
b31_xx_x2_rebar.inp	Second test of Abaqus/Explicit to Abaqus/Explicit import analysis for B31 element.
b32_sx_s_rebar.inp	First test of Abaqus/Standard to Abaqus/Explicit import analysis for B32 element.
b32_sx_x_rebar.inp	Second test of Abaqus/Standard to Abaqus/Explicit import analysis for B32 element.
b32_xs_x_rebar.inp	First test of Abaqus/Explicit to Abaqus/Standard import analysis for B32 element.
b32_xs_s_rebar.inp	Second test of Abaqus/Explicit to Abaqus/Standard import analysis for B32 element.
b32_xx_x1_rebar.inp	First test of Abaqus/Explicit to Abaqus/Explicit import analysis for B32 element.
b32_xx_x2_rebar.inp	Second test of Abaqus/Explicit to Abaqus/Explicit import analysis for B32 element.
p21_sx_s_rebar.inp	First test of Abaqus/Standard to Abaqus/Explicit import analysis for PIPE21 element.
p21_sx_x_rebar.inp	Second test of Abaqus/Standard to Abaqus/Explicit import analysis for PIPE21 element.
p21_xs_x_rebar.inp	First test of Abaqus/Explicit to Abaqus/Standard import analysis for PIPE21 element.
p21_xs_s_rebar.inp	Second test of Abaqus/Explicit to Abaqus/Standard import analysis for PIPE21 element.
p31_sx_s_rebar.inp	First test of Abaqus/Standard to Abaqus/Explicit import analysis for PIPE31 element.
p31_sx_x_rebar.inp	Second test of Abaqus/Standard to Abaqus/Explicit import analysis for PIPE31 element.

p31_xs_x_rebar.inp	First test of Abaqus/Explicit to Abaqus/Standard import analysis for the PIPE31 element.
p31_xs_s_rebar.inp	Second test of Abaqus/Explicit to Abaqus/Standard import analysis for PIPE31 element.

Input files for prestress in beam elements with rebar line reinforcement

b21_rebar_xpl_initc.inp	Prestress test of B21 element.
b22_rebar_xpl_initc.inp	Prestress test of B22 element.
b31_rebar_xpl_initc.inp	Prestress test of B31 element.
b32_rebar_xpl_initc.inp	Prestress test of B32 element.

Input files for mass scaling for beam elements with rebar line reinforcements in Abaqus/Explicit

b21_rebar_xpl_mscf.inp	Abaqus/Explicit fixed mass scaling test for B21 element.
b21_rebar_xpl_mscv.inp	Abaqus/Explicit variable mass scaling test for B21 element.
b22_rebar_xpl_mscf.inp	Abaqus/Explicit fixed mass scaling test for B22 element.
b22_rebar_xpl_mscv.inp	Abaqus/Explicit variable mass scaling test for B22 element.
b31_rebar_xpl_mscf.inp	Abaqus/Explicit fixed mass scaling test for B31 element.
b31_rebar_xpl_mscv.inp	Abaqus/Explicit variable mass scaling test for B31 element.
b32_rebar_xpl_mscf.inp	Abaqus/Explicit fixed mass scaling test for B32 element.
b32_rebar_xpl_mscv.inp	Abaqus/Explicit variable mass scaling test for B32 element.

Figures

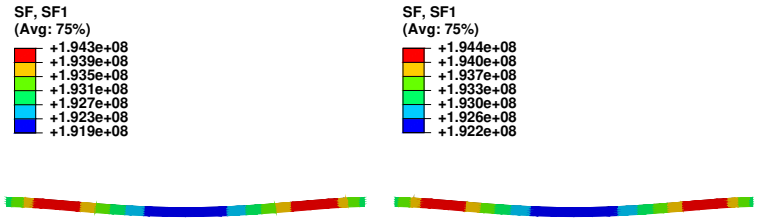


Figure 1: Section force SF1 for a beam structure under uniformly distributed load: Abaqus/Standard result (left) and Abaqus/Explicit result (right).

Convection elements: transport of a temperature pulse

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

DCC1D2 DCC1D2D DCC2D4 DCC2D4D DCC3D8 DCC3D8D DCCAX2 DCCAX2D DCCAX4 DCCAX4D

Features tested

Transport of a temperature pulse in the convective/diffusive elements.

Problem description

The transport and diffusion of a temperature pulse in the family of convective/diffusive heat transfer elements is tested in this verification set. The model consists of a column of fluid 2.0 units long with a cross-sectional area of 1.0. All models consist of 16 elements along the length and one element in the cross-section. The material property values used are: conductivity, $k = 0.0015625$; specific heat, $c_p = 1.0$; and density, $\rho = 1.0$. Consistency of all units is assumed.

An initial temperature pulse, of peak magnitude 1.0, in the form of a Gaussian wave is centered at 0.25 units along the length. At time zero, all the nodes in the model are assigned a mass flow rate of 0.25 in the length direction. The transient response of the temperature pulse as it convects down the length of the mesh is tracked for a period of two seconds.

Results and discussion

The results show that the convective elements are able to propagate a temperature pulse with relatively minor diffusion.

Input files

ec12dch1.inp	DCC1D2 elements.
ec12ddh1.inp	DCC1D2D elements.
ec24dch1.inp	DCC2D4 elements.
ec24ddh1.inp	DCC2D4D elements.
ec38dch1.inp	DCC3D8 elements.
ec38ddh1.inp	DCC3D8D elements.
eca2dch1.inp	DCCAX2 elements.
eca2ddh1.inp	DCCAX2D elements.
eca4dch1.inp	DCCAX4 elements.
eca4ddh1.inp	DCCAX4D elements.

Continuum shells: basic element modes

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

SC8R

Features tested

The basic deformation modes of the continuum shell elements are verified.

Problem description

A continuum shell element is loaded with displacement control into its basic deformation modes. The results are compared to equivalent modes obtained from S4R and C3D8R elements.

Model: The model consists of SC8R, S4R, and C3D8R elements, each of dimensions $2 \times 2 \times 0.1$.

Mesh: Two types of meshes are provided. The mesh for the geometrically linear case consists of three elements: one SC8R, one S4R, and one C3D8R element. Each element is loaded in one of the basic deformation modes. There are 18 steps, each step representing a particular mode. For the geometrically nonlinear case we have 18 groups of SC8R, S4R, and C3D8R elements. Each group is loaded in a particular deformation mode in a single step.

Material: Linear elastic, Young's modulus = 3000.0, Poisson's ratio = 0.0.

Boundary conditions: Each element type is loaded in displacement control to one of the following pure deformation modes: membrane, bending, transverse shear, thickness, thickness gradient, and hourglass.

Results and discussion

The strains, stresses, section strains, section forces, section thicknesses, and reaction forces for the continuum shell are verified with results obtained for the S4R and C3D8R elements for equivalent modes where applicable.

Input files

element_modes_sc8r.inp	Model for geometrically linear case.
element_modes_sc8r_pert.inp	Model for geometrically linear case using linear perturbation steps.
element_modes_sc8r_nlgeom.inp	Model for geometrically nonlinear case.

Transverse shear for shear-flexible shells

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

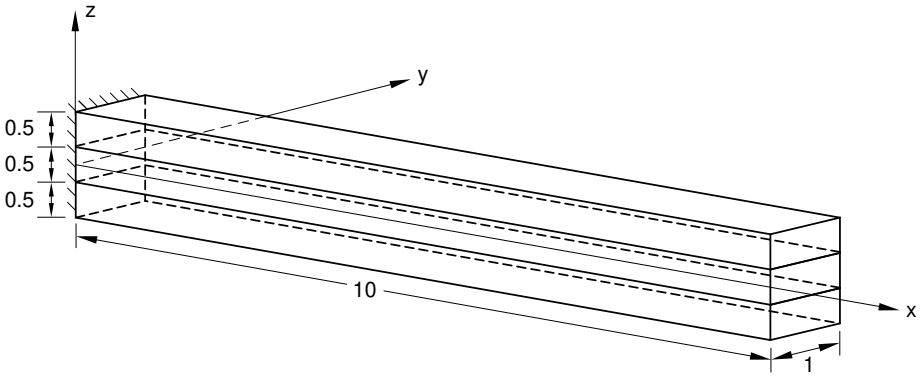
S4 S4R S8R S8RT

Features tested

Transverse shear stress output (TSHR13, TSHR23) and transverse shear section force and section strain output (SF4, SF5, SE4, SE5) for shear-flexible shells.

Problem description

The model consists of a composite plate with a length of 10.0, width of 1.0, and thickness of 1.5. Plane strain conditions are imposed in the *y*-direction (parallel to the unit width), the end at $x = 0$ is fixed, and various boundary conditions are applied to the remaining degrees of freedom (refer to input files). A single shell element is used to model the plate. The plate has three layers of equal thickness (0.5) defined using a composite shell section or a composite general shell section. Three integration points are specified in each layer for a total of nine points through the thickness.



Orthotropic elasticity in a plane stress is used to define an orthotropic material with $E_1 = 25 \times 10^6$, $E_2 = 1 \times 10^6$, $\nu_{12} = 0.25$, $G_{13} = 0.5 \times 10^6$ and $G_{23} = 0.2 \times 10^6$. The material orientation is specified such that the local 1-direction for layers 1 and 3 is parallel to the *x*-axis, and the local 1-direction for layer 2 is parallel to the *y*-axis. A user-defined material that has equivalent behavior to the linear elastic material is also tested.

A section orientation is used with the general shell section tests such that the 1-direction is parallel to the *y*-axis and the 2-direction is parallel and opposite to the *x*-axis. This section orientation only changes local directions for the section forces and section strains.

Gauss integration is used for the shell cross-section for elements S4, S4R, and S8R.

Two groups of tests are performed; all forces are applied at $x = 10$.

Static tests:

Step 1, uniaxial tension: total force of 20000 in the x-direction.

Step 2, transverse shear: total force of 20000 in the z-direction.

Step 3, pure bending: total moment of 20000 about the y-axis.

Static and dynamics tests:

(The first two static steps are performed to correlate (closely) with the eigenmode results of the frequency step.)

Step 1, static, transverse shear: $u_z = 1$ at the $x = 10$ edge.

Step 2, static, uniaxial tension: $u_x = 1$ at the $x = 10$ edge.

Step 3, frequency: extract four lowest eigenmodes.

Step 4, steady-state dynamics: total force of 20000 in the z-direction.

Step 5, steady-state dynamics, direct: total force of 20000 in the z-direction.

Step 6, modal dynamic: total force of 20000 in the z-direction.

Step 7, response spectrum.

Results and discussion

The verification of the transverse shear results is based on the formulation described in *Transverse shear stiffness in composite shells and offsets from the midsurface*.

Local coordinate directions are requested in the input files *esf4sct2.inp*, *esf4slt2.inp*, and *ese4slt2.inp*.

Input files

ese4sct1.inp	S4 elements, orthotropic elastic material, static steps, **SHELL SECTION*, COMPOSITE.
ese4sct1umat.inp	S4 elements, user-defined material, static steps, **SHELL SECTION*, COMPOSITE.
ese4sct2.inp	S4 elements, orthotropic elastic material, static, frequency, steady-state dynamics, modal dynamic, and response spectrum steps, **SHELL SECTION*, COMPOSITE.
ese4sct2umat.inp	S4 elements, user-defined material, static, frequency, steady-state dynamics, modal dynamic, and response spectrum steps, **SHELL SECTION*, COMPOSITE.
ese4slt2.inp	S4 elements, orthotropic elastic material, static, frequency, steady-state dynamics, modal dynamic, and response spectrum steps, **SHELL GENERAL SECTION*, COMPOSITE.
esf4sct1.inp	S4R elements, orthotropic elastic material, static steps, **SHELL SECTION*, COMPOSITE.
esf4sct1umat.inp	S4R elements, user-defined material, static steps, **SHELL SECTION*, COMPOSITE.
esf4sct2.inp	S4R elements, orthotropic elastic material, static, frequency, steady-state dynamics, modal dynamic, and response spectrum steps, **SHELL SECTION*, COMPOSITE.
esf4sct2umat.inp	S4R elements, user-defined material, static, frequency, steady-state dynamics, modal dynamic, and response spectrum steps, **SHELL SECTION*, COMPOSITE.

esf4slt2.inp	S4R elements, orthotropic elastic material, static, frequency, steady-state dynamics, modal dynamic, and response spectrum steps, **SHELL GENERAL SECTION*, COMPOSITE.
es68sct1.inp	S8R elements, orthotropic elastic material, static steps, **SHELL SECTION*, COMPOSITE.
es68sct1umat.inp	S8R elements, user-defined material, static steps, **SHELL SECTION*, COMPOSITE.
es68sct2.inp	S8R elements, orthotropic elastic material, static, frequency, steady-state dynamics, modal dynamic, and response spectrum steps, **SHELL SECTION*, COMPOSITE.
es68sct2umat.inp	S8R elements, user-defined material, static, frequency, steady-state dynamics, modal dynamic, and response spectrum steps, **SHELL SECTION*, COMPOSITE.
es68slt2.inp	S8R elements, orthotropic elastic material, static, frequency, steady-state dynamics, modal dynamic, and response spectrum steps, **SHELL GENERAL SECTION*, COMPOSITE.
es38tct1.inp	S8RT elements, orthotropic elastic material, coupled temperature-displacement steps with static loading, **SHELL SECTION*, COMPOSITE.
es38tct1umat.inp	S8RT elements, user-defined material, coupled temperature-displacement steps with static loading, **SHELL SECTION*, COMPOSITE.
umat_psortho_proptab.f	User subroutine *UMAT* used in ese4sct1umat.inp, ese4sct2umat.inp, esf4sct1umat.inp, esf4sct2umat.inp, es68sct1umat.inp, es68sct2umat.inp, es38tct1umat.inp.

Linear dynamic analysis with fluid link

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

FLINK F2D2

Features tested

Linear dynamic analysis with fluid link elements.

Problem description

A fluid link element is used to transfer fluid between two vessels filled with pneumatic fluid, as shown in *Figure 1*. The vessels are subjected to internal pressures by applying loads F_1 and F_2, respectively.

Each vessel is modeled using a two-dimensional fluid block that measures 1 × 1 with unit thickness as shown in *Figure 2*. Nodes 1 and 11 are the cavity reference nodes for the two fluid cavities. The downward force on the first fluid cavity is applied as a concentrated load to node 4 in the *y*-direction. Nodes 3 and 4 are constrained to displace equally in the *y*-direction. Nodes 13 and 14 are also constrained to displace equally in the *y*-direction. Finally, grounded springs of very small stiffness acting in the *y*-direction are attached to nodes 4 and 14 to prevent solver problems in the solution.

Material:

Pneumatic fluid

Ambient pressure, p_A=14.7.

Absolute zero temperature, θ^Z=–460.

Reference density, ρ_R=10.0.

Reference pressure for density, p_R=0.

Reference temperature for density, θ_R=200.

Initial temperature, θ_I=200.

Fluid link

C_V=10.

Loading: The fluid temperature is kept constant at 200.0 in all of the steps. In the first step, the first cavity is subjected to a concentrated harmonic load of $F_1 = -10.0 \exp[2\pi(0.1)t]$ with $F_2 = 0$. The second step is similar to the first, except that the imaginary terms in the stiffness matrix for the fluid link are ignored, so that the response is calculated only for the real components of the steady-state system. In the third step loads are applied to induce an internal pressure of 10.0 units in both cavities. The fourth and fifth steps are similar to the first and second steps except for the pressure preload of 10.0, which is applied to the fluid elements in the third step. Results are reported at the end of each steady-state analysis step.

Results and discussion

Step	MFL	PHMFL	MFLT	PHMFT	$PCAV_1$	$PPOR_1$
1	1.028	–0.3699	1.635	–90.37	10.28	–0.3699
2					10.28	
4	1.010	–0.2163	1.607	–90.22	10.10	–0.2163
5					10.10	

Input files

efl2sfxd.inp Analysis input file.

Figures

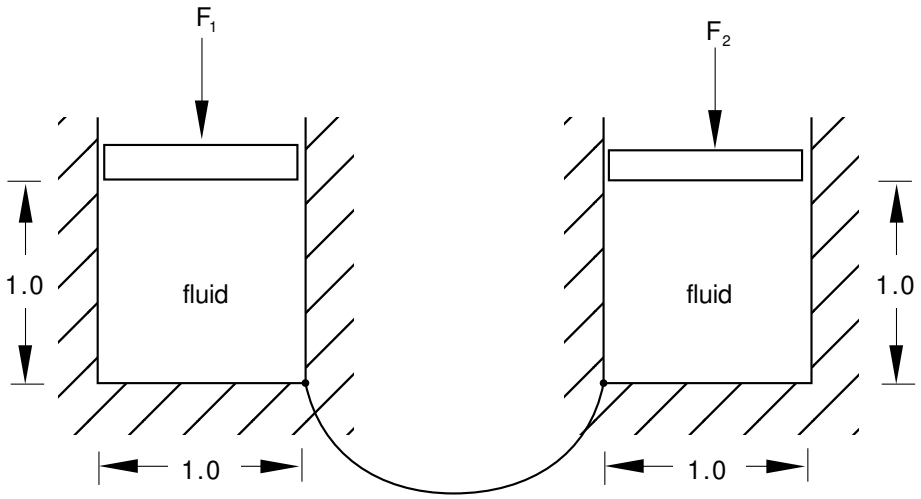


Figure 1: Fluid link model.

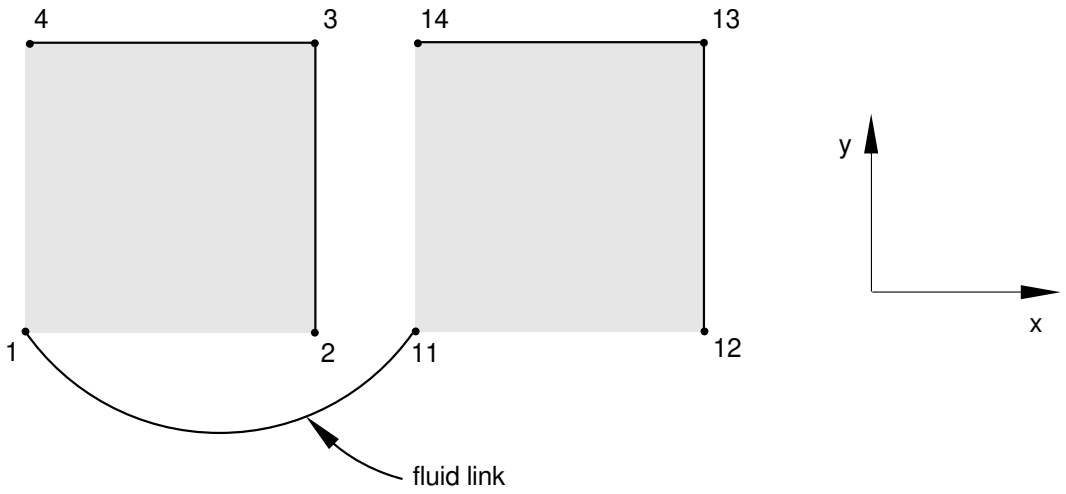


Figure 2: Two-dimensional fluid block model.

Rigid bodies with temperature DOFs, heat capacitance, and nodal-based thermal loads

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Rigid bodies with temperature DOFs

Elements tested

CAX3T CAX4HT CAX4RT CAX4T CAX6MT CAX8HT
CPE3T CPE4RT CPE4T CPE6MT CPE8T
CPS3T CPS4RT CPS4T CPS6MT CPS8T
C3D4T C3D6T C3D8HT C3D8RT C3D8T C3D10MT
SC8RT SC6RT
S3RT S4RT

Problem description

Most of the verification tests in this section are based on the recommendations of the National Agency for Finite Element Methods and Standards (U.K.). Rigid body constraints and isothermal rigid body constraints are tested in these problems.

The test problems are:

1. One-dimensional heat transfer with radiation.
2. One-dimensional transient heat transfer.
3. Two-dimensional heat transfer with convection.
4. Patch test for heat transfer elements.
5. Temperature-dependent film condition.
6. One-element lumped model.

Detailed descriptions of problems (a)–(e) can be found in

- *T2: One-dimensional heat transfer with radiation*;
- *T3: One-dimensional transient heat transfer*;
- *T4: Two-dimensional heat transfer with convection*;
- *Patch test for heat transfer elements*; and
- *Temperature-dependent film condition*, respectively.

The models presented here are the same as the models described in these sections, but the elements are now assigned to rigid bodies.

The one-element lumped model tests the isothermal rigid body constraints. The simulation consists of two steps. In the first step the rigid body is cooled by convection from an initial temperature of T_0=100 to the ambient temperature T_a=20. In the second step the body is heated by a prescribed flux, q. All the thermal properties are equal to unity. In addition to its own thermal capacitance, a second capacitance is lumped into the model using a HEATCAP element.

Results and discussion

The target solutions are reproduced accurately for all the problems tested. For the one-element model the analytical solution is

Step 1:

$$T(t) = T_a + (T_0 - T_a)\,\exp\left[\frac{-hA_1}{(\rho cV)_{total}}\,(t - t_0)\right]$$

Step 2:

$$T(t) = T_a + \frac{qA_2}{(\rho cV)_{total}}\,(t - t_0)$$

In the above equation h is the heat transfer coefficient, $(\rho cV)_{total}$ is the heat capacitance, A_1 is the area associated with the convective flux, t_0 is the time at the end of previous step, and A_2 denotes the area on which the prescribed flux is applied. The temperatures at the nodes are the same because the rigid body is isothermal; therefore, the temperature varies only in time.

In Abaqus/Explicit the internal heat energy ALLIHE and the external heat energy through the external fluxes ALLHF are available. The analytical solutions for the energies are

Step 1:

$$ALLIHE(t) = (\rho cV)_{total}\,T(t),$$

$$ALLHF(t) = -\int_{t_0}^{t} hA_1\,(T - T_a)\,dt = (\rho cV)_{total}\,(T_0 - T_a)\,\left[\exp\left(\frac{-hA_1}{(\rho cV)_{total}}\,(t - t_0)\right) - 1\right]$$

Step 2:

$$ALLIHE(t) = (\rho cV)_{total}\,T(t),$$

$$ALLHF(t) = ALLHF|_{end-step1} + \int_{t_0}^{t} qA_2\,dt = ALLHF|_{end-step1} + qA_2\,(t - t_0)$$

The energies are in good agreement with the analytical solutions, and the heat energy balance is respected.

Input files

Abaqus/Standard input files

One-dimensional heat transfer with radiation:

rbisono_1dhtrd_std_cax4t.inp	CAX4T elements.
rbisono_1dhtrd_std_cps4t.inp	CPS4T elements.
rbisono_1dhtrd_std_c3d8t.inp	C3D8T elements.

One-dimensional transient heat transfer:

rbisono_1dhtcdc_std_cax4t.inp	CAX4T elements, coarse mesh.
rbisono_1dhtcdf_std_cax8ht.inp	CAX8HT elements, fine mesh.
rbisono_1dhtcdc_std_cpe4t.inp	CPE4T elements, coarse mesh.
rbisono_1dhtcdf_std_cpe8t.inp	CPE8T elements, fine mesh.

Two-dimensional heat transfer with convection:

rbisono_2dhtcvc_std_cps4t.inp	CPS4T elements, coarse mesh.
rbisono_2dhtcvf_std_cps8t.inp	CPS8T elements, fine mesh.
rbisono_2dhtcvc_std_c3d8t.inp	C3D8T elements, coarse mesh.

Patch test for heat transfer:

rbisono_htpatch_std_cax4ht.inp CAX4HT elements.
rbisono_htpatch_std_c3d8ht.inp C3D8HT elements.

Temperature-dependent film condition:

rbisono_tempdep_std_cpe4t.inp CPE4T elements.
rbisono_tempdepfm_std_cps4t.inp CPS4T elements and the user subroutine **FILM*.

One-element lumped model:

rbisoyes_heatcap_std_cax4t.inp CAX4T elements.
rbisoyes_heatcap_std_cpe4t.inp CPE4T elements.
rbisoyes_heatcap_std_c3d8t.inp C3D8T elements.

Abaqus/Explicit input files

rbisoyes_heatcap_xpl_sc8rt.inp SC8RT elements.
rbisoyes_heatcap_xpl_s4rt.inp S4RT elements.

One-dimensional heat transfer with radiation:

rbisono_1dhtrd_xpl_cax3t.inp CAX3T elements.
rbisono_1dhtrd_xpl_cax4rt.inp CAX4RT elements.
rbisono_1dhtrd_xpl_cax6mt.inp CAX6MT elements.
rbisono_1dhtrd_xpl_cpe3t.inp CPE3T elements.
rbisono_1dhtrd_xpl_cpe4rt.inp CPE4RT elements.
rbisono_1dhtrd_xpl_cpe6mt.inp CPE6MT elements.
rbisono_1dhtrd_xpl_cps3t.inp CPS3T elements.
rbisono_1dhtrd_xpl_cps4rt.inp CPS4RT elements.
rbisono_1dhtrd_xpl_cps6mt.inp CPS6MT elements.
rbisono_1dhtrd_xpl_c3d4t.inp C3D4T elements.
rbisono_1dhtrd_xpl_c3d6t.inp C3D6T elements.
rbisono_1dhtrd_xpl_c3d8rt.inp C3D8RT elements.
rbisono_1dhtrd_xpl_c3d8t.inp C3D8T elements.

One-dimensional transient heat transfer:

rbisono_1dhtcdc_xpl_cax3t.inp CAX3T elements, coarse mesh.
rbisono_1dhtcdc_xpl_cax4rt.inp CAX4RT elements, coarse mesh.
rbisono_1dhtcdc_xpl_cax6mt.inp CAX6MT elements, coarse mesh.
rbisono_1dhtcdc_xpl_cpe3t.inp CPE3T elements, coarse mesh.
rbisono_1dhtcdc_xpl_cpe4rt.inp CPE4RT elements, coarse mesh.
rbisono_1dhtcdc_xpl_cpe6mt.inp CPE6MT elements, coarse mesh.
rbisono_1dhtcdc_xpl_cps3t.inp CPS3T elements, coarse mesh.
rbisono_1dhtcdc_xpl_cps4rt.inp CPS4RT elements, coarse mesh.
rbisono_1dhtcdc_xpl_cps6mt.inp CPS6MT elements, coarse mesh.
rbisono_1dhtcdc_xpl_s4rt.inp S4RT elements, coarse mesh.
rbisono_1dhtcdf_xpl_cax3t.inp CAX3T elements, fine mesh.
rbisono_1dhtcdf_xpl_cax4rt.inp CAX4RT elements, fine mesh.

rbisono_1dhtcdf_xpl_cpe3t.inp	CPE3T elements, fine mesh.
rbisono_1dhtcdf_xpl_cpe4rt.inp	CPE4RT elements, fine mesh.
rbisono_1dhtcdf_xpl_cps3t.inp	CPS3T elements, fine mesh.
rbisono_1dhtcdf_xpl_cps4rt.inp	CPS4RT elements, fine mesh.
rbisono_1dhtcdf_xpl_s3rt.inp	S3RT elements, fine mesh.

Two-dimensional heat transfer with convection:

rbisono_2dhtcvc_xpl_cpe3t.inp	CPE3T elements, coarse mesh.
rbisono_2dhtcvc_xpl_cpe4rt.inp	CPE4RT elements, coarse mesh.
rbisono_2dhtcvc_xpl_cpe6mt.inp	CPE6MT elements, coarse mesh.
rbisono_2dhtcvc_xpl_cps3t.inp	CPS3T elements, coarse mesh.
rbisono_2dhtcvc_xpl_cps4rt.inp	CPS4RT elements, coarse mesh.
rbisono_2dhtcvc_xpl_cps6mt.inp	CPS6MT elements, coarse mesh.
rbisono_2dhtcvc_xpl_c3d6t.inp	C3D6T elements, coarse mesh.
rbisono_2dhtcvc_xpl_c3d8rt.inp	C3D8RT elements, coarse mesh.
rbisono_2dhtcvc_xpl_c3d8t.inp	C3D8T elements, coarse mesh.
rbisono_2dhtcvc_xpl_c3d6t.inp	C3D6T elements, coarse mesh.
rbisono_2dhtcvf_xpl_cpe3t.inp	CPE3T elements, fine mesh.
rbisono_2dhtcvf_xpl_cpe4rt.inp	CPE4RT elements, fine mesh.
rbisono_2dhtcvf_xpl_cps3t.inp	CPS3T elements, fine mesh.
rbisono_2dhtcvf_xpl_cps4rt.inp	CPS4RT elements, fine mesh.
rbisono_2dhtcvf_xpl_c3d6t.inp	C3D6T elements, fine mesh.
rbisono_2dhtcvf_xpl_c3d8rt.inp	C3D8RT elements, fine mesh.
rbisono_2dhtcvf_xpl_sc8rt.inp	SC8RT elements, fine mesh.

Patch test for heat transfer:

rbisono_htpatch_xpl_cax3t.inp	CAX3T elements.
rbisono_htpatch_xpl_cax4rt.inp	CAX4RT elements.
rbisono_htpatch_xpl_cax6mt.inp	CAX6MT elements.
rbisono_htpatch_xpl_cpe3t.inp	CPE3T elements.
rbisono_htpatch_xpl_cpe4rt.inp	CPE4RT elements.
rbisono_htpatch_xpl_cpe6mt.inp	CPE6MT elements.
rbisono_htpatch_xpl_cps3t.inp	CPS3T elements.
rbisono_htpatch_xpl_cps4rt.inp	CPS4RT elements.
rbisono_htpatch_xpl_cps6mt.inp	CPS6MT elements.
rbisono_htpatch_xpl_c3d4t.inp	C3D4T elements.
rbisono_htpatch_xpl_c3d6t.inp	C3D6T elements.
rbisono_htpatch_xpl_c3d8rt.inp	C3D8RT elements.
rbisono_htpatch_xpl_c3d8t.inp	C3D8T elements.
rbisono_htpatch_xpl_sc8rt.inp	SC8RT elements.

Temperature-dependent film condition:

rbisono_tempdep_xpl_cpe3t.inp	CPE3T elements.
rbisono_tempdep_xpl_cpe4rt.inp	CPE4RT elements.
rbisono_tempdep_xpl_cpe6mt.inp	CPE6MT elements.

rbisono_tempdep_xpl_cps3t.inp	CPS3T elements.
rbisono_tempdep_xpl_cps4rt.inp	CPS4RT elements.
rbisono_tempdep_xpl_cps6mt.inp	CPS6MT elements.
rbisono_tempdep_xpl_s4rt.inp	S4RT elements.

One-element lumped model:

rbisoyes_heatcap_xpl_cax4rt.inp	CAX4RT elements.
rbisoyes_heatcap_xpl_cax6mt.inp	CAX6MT elements.
rbisoyes_heatcap_xpl_cpe4rt.inp	CPE4RT elements.
rbisoyes_heatcap_xpl_cpe6mt.inp	CPE6MT elements.
rbisoyes_heatcap_xpl_cps6mt.inp	CPS6MT elements.
rbisoyes_heatcap_xpl_c3d8rt.inp	C3D8RT elements.
rbisoyes_heatcap_xpl_c3d8t.inp	C3D8T elements.
rbisoyes_heatcap_xpl_c3d10mt.inp	C3D10MT elements.

Heat capacitance

Elements tested

DCAX4 DC2D4 DC2D8 DC3D6 DC3D8 DC3D8
CAX4T CPS4T CPS8RT C3D8T
DCAX4E DC2D4E DC2D8E DC3D8E
CAX4RT CAX6MT CPE4RT CPE6MT CPEG4T CPEG8T CPS6MT C3D8RT C3D8T
C3D10MT SC8RT

Problem description

The test is based on the one-element lumped model described in the previous section.

Results and discussion

The results match the analytical solution.

Input files

Abaqus/Standard input files

heatcapcfilm_std_dcax4.inp	DCAX4 elements.
heatcapcfilm_std_dc2d4.inp	DC2D4 elements.
heatcapcfilm_std_dc2d8.inp	DC2D8 elements.
heatcapcfilm_std_dc3d6.inp	DC3D6 elements.
heatcapcfilm_std_dc3d8.inp	DC3D8 elements.
heatcapcfilm_std_cax4t.inp	CAX4T elements.
heatcapcfilm_std_cpeg4t.inp	CPEG4T elements.
heatcapcfilm_std_cpeg8t.inp	CPEG8T elements.
heatcapcfilm_std_cps4t.inp	CPS4T elements.
heatcapcfilm_std_cps8rt.inp	CPS8RT elements.
heatcapcfilm_std_c3d8t.inp	C3D8T elements.
heatcapcfilm_std_dcax4e.inp	DCAX4E elements.

heatcapcfilm_std_dc2d4e.inp	DC2D4E elements.
heatcapcfilm_std_dc2d8e.inp	DC2D8E elements.
heatcapcfilm_std_dc2d8e_post.inp	**POST OUTPUT* analysis.
heatcapcfilm_std_dc3d8e.inp	DC3D8E elements.

Abaqus/Explicit input files

rbisoyes_heatcap_xpl_cax4rt.inp	CAX4RT elements.
rbisoyes_heatcap_xpl_cax6mt.inp	CAX6MT elements.
rbisoyes_heatcap_xpl_cpe4rt.inp	CPE4RT elements.
rbisoyes_heatcap_xpl_cpe6mt.inp	CPE6MT elements.
rbisoyes_heatcap_xpl_cps6mt.inp	CPS6MT elements.
rbisoyes_heatcap_xpl_c3d8rt.inp	C3D8RT elements.
rbisoyes_heatcap_xpl_c3d8t.inp	C3D8T elements.
rbisoyes_heatcap_xpl_c3d10mt.inp	C3D10MT elements.
rbisoyes_heatcap_xpl_sc8rt.inp	SC8RT elements.

Node-based radiation conditions

Elements tested

DC1D2 DC1D3 DCAX3 DCAX4 DCAX6 DCAX8 DC2D3 DC2D4
DC2D6 DC2D8 DC3D8
CAX8HT CPE4T CPEG4T CPEG8T C3D8HT T2D2T
DCAX6E DC1D2E DC2D3E DC3D8E
CAX3T CAX4RT CPE4RT CPE6MT CPS4RT C3D6T C3D8RT

Problem description

The tests are based on the problem presented in *T2: One-dimensional heat transfer with radiation*. In the tests presented here, element-based radiation conditions are replaced by node-based radiation conditions.

Results and discussion

The results are in good agreement with the target temperature of 653.85°C. For the second-order elements tested in Abaqus/Standard, the radiative loads at the nodes are weighted appropriately to apply consistent nodal loads. For the coupled temperature-displacement and coupled thermal-electrical elements, dummy mechanical and electrical properties are used, respectively, since only the heat transfer analysis is of interest.

Input files

Abaqus/Standard input files

onedht_crad_std_dc1d2.inp	DC1D2 elements.
onedht_crad_std_dc1d3.inp	DC1D3 elements.
onedht_crad_std_dcax3.inp	DCAX3 elements.
onedht_crad_std_dcax4.inp	DCAX4 elements.
onedht_crad_std_dcax6.inp	DCAX6 elements.
onedht_crad_std_dcax8.inp	DCAX8 elements.
onedht_crad_std_dc2d3.inp	DC2D3 elements.

onedht_crad_std_dc2d4.inp	DC2D4 elements.
onedht_crad_std_dc2d6.inp	DC2D6 elements.
onedht_crad_std_dc2d8.inp	DC2D8 elements.
onedht_crad_std_dc3d8.inp	DC3D8 elements.
onedht_crad_std_cax8ht.inp	CAX8HT elements.
onedht_crad_std_cpe4t.inp	CPE4T elements.
onedht_crad_std_cpeg4t.inp	CPEG4T elements.
onedht_crad_std_cpeg8t.inp	CPEG8T elements.
onedht_crad_std_c3d8ht.inp	C3D8HT elements.
onedht_crad_std_t2d2t.inp	T2D2T elements.
onedht_crad_std_dc1d2e.inp	DC1D2E elements.
onedht_crad_std_dc2d3e.inp	DC2D3E elements.
onedht_crad_std_dcax6e.inp	DCAX6E elements.
onedht_crad_std_dc3d8e.inp	DC3D8E elements.

Abaqus/Explicit input files

cradiate_1dhtrd_xpl_cax4rt.inp	CAX4RT elements.
cradiate_1dhtrd_xpl_cpe6mt.inp	CPE6MT elements.
cradiate_1dhtrd_xpl_cpe4rt.inp	CPE4RT elements.
cradiate_1dhtrd_xpl_c3d8rt.inp	C3D8RT elements.

Node-based film conditions and concentrated heat fluxes

Elements tested

DCAX4 DC2D4 DC2D8 DC3D6 DC3D8 CAX3T CPS4RT C3D6T
CAX4T CPS4T CPS8RT C3D8T
DCAX4E DC2D4E DC2D8E DC3D8E
CAX3T CAX6MT CPE6MT CPEG4T CPEG8T CPS4RT CPS6MT C3D6T C3D10MT SC6RT

Problem description

The tests are based on the one-element lumped model described earlier. The nodal thermal loads node-based film conditions and concentrated heat fluxes are used for cooling and heating the body, respectively. As with the node-based radiation conditions tests described earlier, in Abaqus/Standard the nodal loads are weighted appropriately for the second-order elements; dummy mechanical and electrical properties are used for the coupled temperature-displacement and coupled thermal-electrical analyses, respectively.

Results and discussion

The temperature values are in good agreement with the analytical solution.

Input files

Abaqus/Standard input files

heatcapcfilm_std_dcax4.inp	DCAX4 element.
heatcapcfilm_std_dc2d4.inp	DC2D4 element.

heatcapcfilm_std_dc2d8.inp	DC2D8 element.
heatcapcfilm_std_dc3d6.inp	DC3D6 element.
heatcapcfilm_std_dc3d8.inp	DC3D8 element.
heatcapcfilm_std_cax4t.inp	CAX4T element.
heatcapcfilm_std_cpeg4t.inp	CPEG4T elements.
heatcapcfilm_std_cpeg8t.inp	CPEG8T elements.
heatcapcfilm_std_cps4t.inp	CPS4T element.
heatcapcfilm_std_cps8rt.inp	CPS8RT element.
heatcapcfilm_std_c3d8t.inp	C3D8T element.
heatcapcfilm_std_dcax4e.inp	DCAX4E element.
heatcapcfilm_std_dc2d4e.inp	DC2D4E element.
heatcapcfilm_std_dc2d8e.inp	DC2D8E element.
heatcapcfilm_std_dc3d8e.inp	DC3D8E element.

Abaqus/Explicit input files

cfilm_cflux_xpl_cax3t.inp	CAX3T element.
cfilm_cflux_xpl_cax6mt.inp	CAX6MT element.
cfilm_cflux_xpl_cpe6mt.inp	CPE6MT element.
cfilm_cflux_xpl_cps4rt.inp	CPS4RT element.
cfilm_cflux_xpl_cps6mt.inp	CPS6MT element.
cfilm_cflux_xpl_c3d6t.inp	C3D6T element.
cfilm_cflux_xpl_c3d10mt.inp	C3D10MT element.
cfilm_cflux_xpl_sc6rt.inp	SC6RT element.

Thermal contact between rigid bodies

Elements tested

CPE4T CPS4T
CPE4RT CPS4RT CPE6MT

Problem description

The tests are based on the problems presented in *Thermal surface interaction* and *Coupled temperature-displacement analysis: one-dimensional gap conductance and radiation*. In the first set of tests only the temperature variation in the rigid bodies involved in contact is considered, since the deformations are not of interest. In Abaqus/Explicit two types of thermal contact are considered: thermal contact between a rigid body and an analytical rigid surface and thermal contact between two rigid bodies.

The second test is done in Abaqus/Standard to test the friction dependency on field variables. The test is described in *Coupled temperature-displacement analysis: one-dimensional gap conductance and radiation*; however, here we release the constraints in the tangential direction of contact.

Results and discussion

The temperature values match the results obtained with deformable elements for the first set of tests. In the second set of tests the results obtained using the field variable-dependent friction agree exactly with the results obtained without field variable dependence.

Input files

Abaqus/Standard input files

rb_rb_thcontactc_std_cpe4t.inp	CPE4T elements as rigid bodies; **GAP CONDUCTANCE* test.
rb_rb_thcontactr_std_cps4t.inp	CPS4T elements as rigid bodies; **GAP RADIATION* test.
field_contactp_std_cps4t.inp	CPS4T elements, with field variable-dependent friction; pressure-dependent **GAP CONDUCTANCE*.
field_contactp_std_cps4t_po.inp	**POST OUTPUT* analysis.
nofield_contactp_std_cps4t.inp	CPS4T elements, without field variable-dependent friction; pressure-dependent **GAP CONDUCTANCE*.

Abaqus/Explicit input files

rb_ar_thcontactc_xpl_cps4rt.inp	CPS4RT elements and an analytical rigid surface; **GAP CONDUCTANCE* test.
rb_rb_thcontactc_xpl_cpe4rt.inp	CPE4RT elements as rigid bodies; **GAP CONDUCTANCE* test.
rb_rb_thcontactc_xpl_cpe6mt.inp	CPE6MT elements as rigid bodies; **GAP CONDUCTANCE* test.
rb_ar_thcontactr_xpl_cpe4rt.inp	CPE4RT elements and an analytical rigid surface; **GAP RADIATION* test.
rb_ar_thcontactr_xpl_cpe6mt.inp	CPE6MT elements and an analytical rigid surface; **GAP RADIATION* test.
rb_rb_thcontactr_xpl_cps4rt.inp	CPS4RT elements as rigid elements; **GAP RADIATION* test.

Analysis of unbounded acoustic regions

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Analysis of an acoustic duct

Elements tested

ACIN2D2 ACIN2D3 ACIN3D3 ACIN3D4 ACIN3D6 ACIN3D8 ACINAX2 ACINAX3

Features tested

Steady-state and transient dynamic analysis using acoustic infinite elements.

Problem description

The problem of propagation of plane waves in a duct is used to verify the behavior of acoustic infinite elements. The duct is 10 units long and is excited at one end. The duct itself is modeled with acoustic finite elements of appropriate dimension and interpolation order. At the opposite end acoustic infinite elements are used to simulate the infinite continuation of the duct. In each input file another duct model using the exact plane-wave absorbing impedance boundary condition is supplied for comparison. Although the infinite elements are not exact for the duct case, they should give comparable results to the plane wave impedance case.

The axisymmetric elements are studied using an annular duct terminated with axisymmetric acoustic infinite elements. The comparison duct is identical but oriented in the opposite direction and terminated with the plane wave impedance condition.

Material: Acoustic fluid:

Acoustic bulk modulus, K_f=1.28E5

Acoustic mass density, ρ_f=1.25

Speed of sound, c_0=320.0.

Loading: In the Abaqus/Standard verification files, a two-step analysis is performed. In the first step a steady-state dynamic analysis is performed at two frequencies: 1 and 10. In the second step the fluid in the duct is initially quiescent and is forced at one end using a uniform sinusoidal excitation at a frequency of $5/\pi$. In every case except the ACIN3D6 and ACIN3D8 verification files, the excitation is supplied as a concentrated load; for ACIN3D6 and ACIN3D8 the excitation is supplied as a boundary condition. The reference solution is found using an identical acoustic finite element mesh, with the plane wave impedance condition applied.

In the Abaqus/Explicit verification files, a single-step transient dynamic analysis is performed. The fluid in the duct is initially quiescent and is forced at one end using a uniform sinusoidal excitation at a frequency of $5/\pi$. In every case the excitation is supplied using the concentrated load procedure. The reference solution is found using an identical acoustic finite element mesh, with the plane wave impedance condition applied.

Results and discussion

In each step the solutions using infinite elements produce results comparable to those obtained in the companion plane wave impedance case.

Input files

Abaqus/Standard input files

duct_acin2d2.inp	Duct mesh made up of AC2D4 elements, terminated with an ACIN2D2 element.
duct_acin2d3.inp	Duct mesh made up of AC2D8 elements, terminated with an ACIN2D3 element.
duct_acin3d3.inp	Duct mesh made up of AC3D6 elements, terminated with an ACIN3D3 element.
duct_ac3d5_acin3d4.inp	Duct mesh made up of AC3D5 elements, terminated with an ACIN3D4 element.
duct_acin3d4.inp	Duct mesh made up of AC3D8 elements, terminated with an ACIN3D4 element.
duct_acin3d6.inp	Duct mesh made up of AC3D10 elements, terminated with ACIN3D6 elements.
duct_acin3d8.inp	Duct mesh made up of AC3D20 elements, terminated with an ACIN3D8 element.
duct_acinax2.inp	Duct mesh made up of ACAX4 elements, terminated with an ACINAX2 element.
duct_acinax3.inp	Duct mesh made up of ACAX8 elements, terminated with an ACINAX3 element.

Abaqus/Explicit input files

duct_acin2d2_xpl.inp	Duct mesh made up of AC2D4R elements, terminated with an ACIN2D2 element.
duct_acin3d3_xpl.inp	Duct mesh made up of AC3D8R elements, terminated with an ACIN3D3 element.
duct_acin3d4_xpl.inp	Duct mesh made up of AC3D8R elements, terminated with an ACIN3D4 element.
duct_acinax2_xpl.inp	Duct mesh made up of ACAX4R elements, terminated with an ACINAX2 element.

Coupling to solid elements

Elements tested

ACIN2D2 ACIN2D3 ACIN3D3 ACIN3D4 ACIN3D6 ACIN3D8 ACINAX2 ACINAX3

Problem description

A simple transient problem is studied to verify the coupling of acoustic infinite elements directly to structural elements. Acoustic infinite elements are coupled to solid elements using a surface-based tie constraint. Accelerations are imposed on the solid elements using boundary conditions. To check these results, similar

acceleration profiles are imposed as concentrated loads on acoustic infinite elements of the same geometry. The acceleration time histories are described using the amplitude procedure.

Results and discussion

The time histories for acoustic pressure are in agreement for the two cases for the elements tested. There is a small numerical difference in the method in which accelerations and loads are imposed in Abaqus, which accounts for the small differences observed.

Input files

Abaqus/Standard input files

surf_acin2d2.inp	ACIN2D2 element.
surf_acin2d3.inp	ACIN2D3 element.
surf_acin3d3.inp	ACIN3D3 element.
surf_acin3d4.inp	ACIN3D4 element.
surf_acin3d6.inp	ACIN3D6 element.
surf_acin3d8.inp	ACIN3D8 element.
surf_acinax2.inp	ACINAX2 element.
surf_acinax3.inp	ACINAX3 element.

Abaqus/Explicit input files

surf_acin2d2_xpl.inp	ACIN2D2 element.
surf_acin3d3_xpl.inp	ACIN3D3 element.
surf_acin3d4_xpl.inp	ACIN3D4 element.
surf_acinax2_xpl.inp	ACINAX2 element.

One-dimensional steady-state dynamic solutions for poroelastic acoustic elements

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Problem with analytical solution

Elements tested

C3D4A C3D6A C3D8A

Problem description

A column of poroelastic material fixed in the lateral directions and attached to a wall is loaded at the tip by a harmonic pressure in a direct steady-state dynamic analysis. The analytical solution for the two compressional waves is available in *Allard and Atalla (2009)*, sec. 6.5–6.6. Finite element solutions are obtained for three models having 10, 100, and 1000 finite elements in this one-dimensional problem to verify convergence to the analytical solution. To verify tie constraints to connect poroelastic-to-poroelastic elements, each model is subdivided into two parts with the same material.

Material: The skeleton is linear poroelastic with:

Porosity, 0.94

Tortuosity, 1.06

Young's modulus, 4.4×10^{6}

Poisson's ratio, 0

Density, 130

Structural material bulk modulus, 1.0×10^{10}

Pore fluid, air

Density, 1.177

Static flow resistivity, 40000

Viscous characteristic length, 56×10^{-6}

Thermal characteristic length, 110×10^{-6}

Fluid dynamic shear viscosity, 1.846×10^{-5}

Ambient fluid standard pressure, 1.0×10^{5}

Ambient fluid heat capacity ratio, 1.4

Prandtl number, 0.707

The MKS unit system is used. These parameters are used directly with the Biot-Johnson acoustic medium porous model. For the Biot-Atalla model the equivalent fluid complex density and fluid complex bulk modulus are provided instead.

Boundary conditions: The mesh is aligned along the *Z*-direction. Lateral *X*- and *Y*-displacements are applied at all nodes, and all three displacements at the wall nodes are fixed. External pressure is specified as a boundary condition for the pore pressure and as an external pressure load at the tip element face.

Results and discussion

A solution is obtained for several excitation frequencies in accordance with *Allard and Atalla (2009)*. With mesh refinement, asymptotic pointwise convergence to the analytical solution is $O\left(\Delta x^2\right)$ for the tip displacements and pore pressure at the wall in both real and imaginary components. Results for the C3D8A elements for two excitation frequencies, 300 and 1300 Hz, are presented below.

Real tip displacements, 300 Hz, analytical solution = -2.878E-08	
Number of elements	**Error**
10	-5.94E-11
100	-5.92E-13
1000	-5.E-15

Imaginary tip displacements, 300 Hz, analytical solution = -8.784E-09	
Number of elements	**Error**
10	-6.07E-11
100	-6.09E-13
1000	-6.5E-15

Real pore pressure at the wall, 300 Hz, analytical solution = -7.765E-02	
Number of elements	**Error**
10	-2.14E-03
100	-2.14E-05
1000	-2.11E-07

Imaginary pore pressure at the wall, 300 Hz, analytical solution = -2.768E-01	
Number of elements	**Error**
10	-9.44E-04
100	-9.59E-06
1000	-1.1E-07

Real tip displacements, 1300 Hz, analytical solution = -1.077E-08	
Number of elements	**Error**
10	1.77E-09
100	1.51E-11
1000	1.53E-13

Imaginary tip displacements, 1300 Hz, analytical solution = -6.522E-09	
Number of elements	**Error**
10	3.79E-10

Imaginary tip displacements, 1300 Hz, analytical solution = -6.522E-09	
Number of elements	**Error**
100	3.19E-12
1000	3.25E-14

Real pore pressure at the wall, 1300 Hz, analytical solution = -6.516E-02	
Number of elements	**Error**
10	5.13E-03
100	4.32E-05
1000	4.36E-07

Imaginary pore pressure at the wall, 1300 Hz, analytical solution = 6.955E-03	
Number of elements	**Error**
10	8.41E-03
100	7.84E-05
1000	7.92E-07

Input files

aa2009anal.inp	C3D8A elements, 10 element mesh.
aa2009anal2.inp	C3D8A elements, 100 element mesh.
aa2009anal3.inp	C3D8A elements, 1000 element mesh.
aa2009analc3d6a.inp	C3D6A elements, 20 element mesh (each cube of the 10 element mesh is subdivided into two wedges).
aa2009analc3d6a2.inp	C3D6A elements, 200 element mesh (each cube of the 100 element mesh is subdivided into two wedges).
aa2009analc3d6a3.inp	C3D6A elements, 2000 element mesh (each cube of the 1000 element mesh is subdivided into two wedges).
aa2009analc3d4a.inp	C3D4A elements, 60 element mesh (each cube of the 10 element mesh is subdivided into six tetrahedra).
aa2009analc3d4a2.inp	C3D4A elements, 600 element mesh (each cube of the 100 element mesh is subdivided into six tetrahedra).
aa2009analc3d4a3.inp	C3D4A elements, 6000 element mesh (each cube of the 1000 element mesh is subdivided into six tetrahedra).
aa2009analconstr2.inp	C3D8A elements, 10 element mesh (5 plus 5 matched) with tie constraints.
aa2009analconstr4.inp	C3D8A elements, 15 element mesh (10 plus 5 unmatched) with tie constraints.

References

- Allard, A. F., and N. Atalla, Propagation of Sound in Porous Media: Modelling Sound Absorbing Materials, John Wiley & Sons, Ltd., Second Edition, 2009.

Testing compatibility boundary conditions

Problem description

The problem is similar to the problem above but does not need an analytical solution. It tests poroelastic-to-elastic and poroelastic-to-acoustic compatibility boundary conditions. An acoustic problem is first solved with acoustic AC3D8 elements, and pore pressure results should coincide with the results of a combined AC3D8 plus C3D8A (with porosity close to 1) element model. An elastic problem is first solved with solid C3D8 elements, and displacement and stress results should coincide with the results of a combined C3D8 plus C3D8A (with zero porosity) element model.

Results and discussion

The results agree well with the results from the reference solutions.

Input files

ac3d8_c3d8aconstr4.inp	AC3D8 elements, 10 element mesh, to be compared against the results of ac3d8_c3d8aconstr5.inp.
ac3d8_c3d8aconstr5.inp	AC3D8 plus C3D8A elements, 15 element mesh, to be compared against the results of ac3d8_c3d8aconstr4.inp.
c3d8_c3d8aconstr3.inp	C3D8 elements, 10 element mesh, to be compared against the results of c3d8_c3d8aconstr5.inp.
c3d8_c3d8aconstr5.inp	C3D8 plus C3D8A elements, 15 element mesh, to be compared against the results of c3d8_c3d8aconstr3.inp.

Nonstructural mass verification

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Various methods for including a nonstructural mass in a model are tested. Most of the analyses consist of a set of reference elements that do not include a nonstructural mass and another set of test elements whose material density and nonstructural mass contribution are adjusted to make the total mass equal those of the reference elements. The response of the test elements should be identical to that of the reference elements.

Nonstructural mass specified in the form of a total mass over a region of uniform material density

Elements tested

B21 B22 B31 B32 PIPE21 PIPE31
C3D4 C3D6 C3D8 C3D8R SC6R SC8R
CAX3 CAX4R
CPE3 CPE4R CPS3 CPS4R
M3D3 M3D4R
S3R S4 S4R SAX1
T2D2 T3D2

Problem description

The nonstructural mass contribution is specified in the form of a total mass to be applied over an element set. Several element types are tested in each input file, with two elements in the model for each element type. Each element pair is subjected to equivalent displacements (and rotations in the case of beams and shells) such that their response is dynamic. Tests of membranes and shells are performed with and without the specification of nodal thickness. The reaction forces for the constrained nodes of each pair of elements are output for comparison purposes.

Results and discussion

Reaction force histories for nodes on each pair of test and reference elements are nearly identical.

Input files

std_nsm_tot_continuum.inp	Abaqus/Standard analysis of two-dimensional and three-dimensional continuum elements.
std_nsm_tot_beamshell.inp	Abaqus/Standard analysis of two-dimensional and three-dimensional beams, pipes, and shells.
xpl_nsm_tot_continuum.inp	Abaqus/Explicit analysis of two-dimensional and three-dimensional continuum elements.
xpl_nsm_tot_beamshell.inp	Abaqus/Explicit analysis of two-dimensional and three-dimensional beams and shells.

Nonstructural mass specified in the form of a total mass over a region of nonuniform material density

Elements tested

B21 B22 B31 B32
C3D4 C3D6 C3D8 C3D8R SC6R SC8R
CAX3 CAX4R
CPE3 CPE4R CPS3 CPS4R
M3D3 M3D4R
S3R S4 S4R SAX1
T2D2 T3D2

Problem description

The nonstructural mass contribution is specified in the form of a total mass to be applied over the entire model. Several element types are tested in each input file with two elements (test and reference) in the model for each element type. The material density of a "reference" element is chosen to be eight times that of a "test" element. A total mass equal to a third of all "reference" elements is distributed over the entire model. In the case of a mass proportional distribution of the nonstructural mass, the effective element densities of a "reference" element and a "test" element remain at the 8:1 ratio; with the volume proportional distribution, the ratio changes to 4:1. In either distribution any "test" and "reference" element pair would have different mass; hence, the reaction forces are not expected to match.

In Abaqus/Explicit each element pair is subjected to equivalent displacements (and rotations in the case of beams and shells) such that their response is dynamic. In Abaqus/Standard a single- step static analysis is carried out with gravity loads. Rebar defined using the procedure to specify layers of reinforcement as part of the element section definition are included where applicable. Under mass proportional distribution of a total nonstructural mass, the elements with rebar defined using the procedure to specify layers of reinforcement as part of the element section definition attract a higher nonstructural mass compared to those elements without the rebar. However, the same is not true when the rebar are defined using the element-based rebar procedure. Tests of membranes and shells are performed with and without the specification of nodal thickness. The reaction forces for the constrained nodes of each pair of elements are output. In Abaqus/Explicit the element stable time increment values are also output for comparison. These values for a "reference" element and a "test" element are not expected to be identical but should correspond to the modified spatial distribution of the mass in the model.

Results and discussion

The masses of each "reference" and "test" element pair are output to the printed output (`.dat`) file using a model definition data request. The results match the expected values.

Input files

std_nsm_tot_continuum_m.inp	Abaqus/Standard analysis of two-dimensional and three-dimensional continuum elements with mass proportional distribution of the nonstructural mass.
std_nsm_tot_beamshell_m.inp	Abaqus/Standard analysis of two-dimensional and three-dimensional beams and shells with mass proportional distribution of the nonstructural mass.
std_nsm_tot_continuum_v.inp	Abaqus/Standard analysis of two-dimensional and three-dimensional continuum elements with volume proportional distribution of the nonstructural mass.
std_nsm_tot_beamshell_v.inp	Abaqus/Standard analysis of two-dimensional and three-dimensional beams and shells with volume proportional distribution of the nonstructural mass.

xpl_nsm_tot_continuum_m.inp	Abaqus/Explicit analysis of two-dimensional and three-dimensional continuum elements with mass proportional distribution of the nonstructural mass.
xpl_nsm_tot_beamshell_m.inp	Abaqus/Explicit analysis of two-dimensional and three-dimensional beams and shells with mass proportional distribution of the nonstructural mass.
xpl_nsm_tot_continuum_v.inp	Abaqus/Explicit analysis of two-dimensional and three-dimensional continuum elements with volume proportional distribution of the nonstructural mass.
xpl_nsm_tot_beamshell_v.inp	Abaqus/Explicit analysis of two-dimensional and three-dimensional beams and shells with volume proportional distribution of the nonstructural mass.

Nonstructural mass specified in the form of a mass per unit volume

Elements tested

B21 B22 B31 B32
C3D4 C3D6 C3D8 C3D8R SC6R SC8R
CAX3 CAX4R
CPE3 CPE4R CPS3 CPS4R
M3D3 M3D4R
S3R S4 S4R SAX1
T2D2 T3D2

Problem description

The nonstructural mass contribution is specified in the form of a mass per unit volume to be applied over an element set. Several element types are tested in each input file, with two elements in the model for each element type. Each element pair is subjected to equivalent displacements (and rotations in the case of beams and shells) such that their response is dynamic. Tests of membranes and shells are performed with and without the specification of nodal thickness. The reaction forces for the constrained nodes of each pair of elements are output for comparison purposes.

Results and discussion

Reaction force histories for nodes on each pair of test and reference elements are nearly identical.

Input files

std_nsm_mpv_continuum.inp	Abaqus/Standard analysis of two-dimensional and three-dimensional continuum elements.
std_nsm_mpv_beamshell.inp	Abaqus/Standard analysis of two-dimensional and three-dimensional beams and shells.
xpl_nsm_mpv_continuum.inp	Abaqus/Explicit analysis of two-dimensional and three-dimensional continuum elements.
xpl_nsm_mpv_beamshell.inp	Abaqus/Explicit analysis of two-dimensional and three-dimensional beams and shells.

Nonstructural mass specified in the form of a mass per unit area

Elements tested

M3D3 M3D4R
S3R S4 S4R SAX1

Problem description

The nonstructural mass contribution is specified in the form of a mass per unit area to be applied over an element set. Several element types are tested in each input file, with two elements in the model for each element type. Each element pair is subjected to equivalent displacements (and rotations in the case of beams and shells) such that their response is dynamic. Tests of membranes and shells are performed with and without the specification of nodal thickness. The reaction forces for the constrained nodes of each pair of elements are output for comparison purposes.

Results and discussion

Reaction force histories for nodes on each pair of test and reference elements are nearly identical.

Input files

std_nsm_mpa_continuum.inp	Abaqus/Standard analysis of two-dimensional and three-dimensional continuum elements.
std_nsm_mpa_beamshell.inp	Abaqus/Standard analysis of two-dimensional and three-dimensional beams and shells.
xpl_nsm_mpa_continuum.inp	Abaqus/Explicit analysis of two-dimensional and three-dimensional continuum elements.
xpl_nsm_mpa_beamshell.inp	Abaqus/Explicit analysis of two-dimensional and three-dimensional beams and shells.

Nonstructural mass specified in the form of a mass per unit length

Elements tested

B21 B22 B31 B32 PIPE21 PIPE31
T2D2 T3D2

Problem description

The nonstructural mass contribution is specified in the form of a mass per unit length to be applied over an element set. Several element types are tested in each input file, with two elements in the model for each element type. Each element pair is subjected to equivalent displacements (and rotations in the case of beams and shells) such that their response is dynamic. The reaction forces for the constrained nodes of each pair of elements are output for comparison purposes.

Results and discussion

Reaction force histories for nodes on each pair of test and reference elements are nearly identical.

Input files

std_nsm_mpl_continuum.inp	Abaqus/Standard analysis of two-dimensional and three-dimensional continuum elements.
std_nsm_mpl_beamshell.inp	Abaqus/Standard analysis of two-dimensional and three-dimensional beams and shells.
xpl_nsm_mpl_continuum.inp	Abaqus/Explicit analysis of two-dimensional and three-dimensional continuum elements.
xpl_nsm_mpl_beamshell.inp	Abaqus/Explicit analysis of two-dimensional and three-dimensional beams, pipes, and shells.

Mass adjust verification

Products: Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Use of mass adjust to define the total mass of an element set

Elements tested

B21 B22 B31 B32 PIPE21 PIPE31
C3D4 C3D6 C3D8 C3D8R SC6R SC8R
CAX3 CAX4R
CPE3 CPE4R CPS3 CPS4R
M3D3 M3D4R
S3R S4 S4R SAX1
T2D2 T3D2

Problem description

Using mass adjustment to define the total mass of an element set in a model is tested. The analyses consist of a set of reference elements and another set of test elements whose material density is different from that of the reference elements. Mass adjustment is applied to the test elements to make the total mass equal those of the reference elements. The response of the test elements should be identical to that of the reference elements. Several element types are tested in each input file, with two elements in the model for each element type. Each element pair is subjected to equivalent displacements (and rotations in the case of beams and shells) such that their response is dynamic. Tests of membranes and shells are performed with and without the specification of nodal thickness. The reaction forces for the constrained nodes of each pair of elements are output for comparison purposes.

Results and discussion

Reaction forces for nodes on each pair of test and reference elements are nearly identical.

Input files

massadjust_tot_continuum.inp	Abaqus/Explicit analysis of two-dimensional and three-dimensional continuum elements.
massadjust_tot_beamshell.inp	Abaqus/Explicit analysis of two-dimensional and three-dimensional beams and shells.

Use of mass adjust to redistribute element masses to satisfy the minimum element stable time increment requirement

Elements tested

C3D8 C3D8R S3R S4R

Problem description

Using mass adjustment to define the total mass of an element set is tested, with redistribution of the element masses to satisfy the minimum stable element time increment requirement. A circular plate subject to blast loading is considered for the test. Considering symmetry, a quarter of the plate is meshed with square elements in an inner region and non-square elements in the outer region. The elements in the inner region are smaller in size than those in the outer region and have a lower material density. In a reference analysis, fixed mass scaling is specified and the mass addition in the inner region due to mass scaling is noted. There is no mass scaling in the outer region. In the test, first the mass of the outer region is adjusted to include the additional mass of the inner region noted in the reference analysis. Next, the mass of the whole model is adjusted to the total mass (including mass scaling) of the whole model of the reference, with the same minimum element stable time increment used for mass scaling. Thus, in the test analysis, the mass added to the outer region from the first mass adjust data entry will be redistributed in its entirety to the inner region as a result of the second. The resulting masses of the inner and outer regions in the test should be identical to those in the reference. The dynamic response of the test should also be the same as the reference.

Results and discussion

The masses of the element sets are output to the printed output (`.dat`) file using a model definition data request. The results of the test match those of the reference analysis.

Input files

massadjust_dt_c3d8_plate.inp	Abaqus/Explicit analysis using C3D8 elements and mass adjustment to impose minimum stable element time increment.
massadjust_dt_c3d8_plate_ref.inp	Abaqus/Explicit analysis using C3D8 elements and fixed mass scaling to impose minimum stable element time increment.
massadjust_dt_c3d8r_plate.inp	Abaqus/Explicit analysis using C3D8R elements and mass adjustment to impose minimum stable element time increment.
massadjust_dt_c3d8r_plate_ref.inp	Abaqus/Explicit analysis using C3D8R elements and fixed mass scaling to impose minimum stable element time increment.
massadjust_dt_s3r_plate.inp	Abaqus/Explicit analysis using S3R elements and mass adjustment to impose minimum stable element time increment.
massadjust_dt_s3r_plate_ref.inp	Abaqus/Explicit analysis using S3R elements and fixed mass scaling to impose minimum stable element time increment.
massadjust_dt_s4r_plate.inp	Abaqus/Explicit analysis using S4R elements and mass adjustment to impose minimum stable element time increment.
massadjust_dt_s4r_plate_ref.inp	Abaqus/Explicit analysis using S4R elements and fixed mass scaling to impose minimum stable element time increment.

Defining a reference mesh

Products: Abaqus/Explicit

References:

- *Defining a Reference Mesh for Membrane Elements and Three-Dimensional Solid Elements*

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D5 C3D6 C3D8 C3D8I C3D8R

Problem description

If a reference mesh is specified for an element, Abaqus/Explicit computes the initial stresses based on the deformation gradient that maps the reference configuration to the initial configuration. This capability is tested for different element types and materials.

Some tests consist of a two-analysis sequence. In the first analysis, a regular Abaqus/Explicit simulation is performed to compute the element stresses resulting from applied loads and boundary conditions. In the second analysis, the initial mesh of the first analysis is used as the reference mesh and the final configuration of the first analysis is used as the initial mesh. This is accomplished by adjusting the user-specified nodal coordinates (see *Adjusting Nodal Coordinates*) using the displacement field data from the first analysis (*Using an external field to define distributions and initial conditions*).

Results and discussion

In all cases, the initial stresses reported at the zero increment match the expected results. This verifies that the definition of a reference mesh leads to the correct initial stress conditions in each case.

Several test cases use the two-analysis sequence with different element types (C3D8 and C3D8R, C3D5 and C3D6). As shown in *Figure 1* (an indentation test using C3D8R elements with elastic material), the initial stresses of the second analysis (right image) computed based on the reference mesh method are very close to the stresses in the final configuration of the first analysis (left image). Additional test cases include a test with three brick elements of different element types (C3D8, C3D8I, and C3D8R) using the same hyperelastic material and reference mesh and two tests consisting of four C3D8 or C3D8R elements each with a reference mesh of different spatial rotations. In each case, it is verified that the initial stresses computed based on the reference mesh method are correct.

Input files

inimetric-indent-ela-c3d8r-0.inp Indentation test for C3D8R elements using elastic material. First analysis of two-analysis sequence. Displacement results from SIM output used to compute initial mesh for second analysis, `inimetric-indent-ela-c3d8r-1.inp`.

inimetric-indent-ela-c3d8r-1.inp Second analysis of two-analysis sequence. Reference mesh defined as initial mesh of first analysis, `inimetric-indent-ela-c3d8r-0.inp`.

inimetric-indent-hyper-c3d8r-0.inp Indentation test for C3D8R elements using hyperelastic material. First analysis of two-analysis sequence. Displacement results from SIM output used to compute initial mesh for second analysis, `inimetric-indent-hyper-c3d8r-1.inp`.

inimetric-indent-hyper-c3d8r-1.inp Second analysis of two-analysis sequence. Reference mesh defined as initial mesh of first analysis, `inimetric-indent-hyper-c3d8r-0.inp` .

inimetric-indent-hyper-c3d8-0.inp Indentation test for C3D8 elements using hyperelastic material. First analysis of two-analysis sequence. Displacement results from SIM output used to compute initial mesh for second analysis, `inimetric-indent-hyper-c3d8-1.inp`.

inimetric-indent-hyper-c3d8-1.inp Second analysis of two-analysis sequence. Reference mesh defined as initial mesh of first analysis, `inimetric-indent-hyper-c3d8-0.inp`.

inimetric-c3d5-0.inp Reference mesh test for C3D5 elements using hyperelastic material. First analysis of two-analysis sequence.

inimetric-c3d5-1.inp Second analysis of two-analysis sequence. Reference mesh defined as initial mesh of first analysis, `inimetric-c3d5-0.inp`.

inimetric-c3d6-0.inp Reference mesh test for C3D6 elements using hyperelastic material. First analysis of two-analysis sequence.

inimetric-c3d6-1.inp Second analysis of two-analysis sequence. Reference mesh defined as initial mesh of first analysis, `inimetric-c3d6-0.inp`.

inimetric-brick-hyper.inp Test with three different types of brick elements (C3D8, C3D8I, and C3D8R) using the same hyperelastic material and reference mesh.

inimetric-c3d8r-ela.inp Test with four C3D8R elements, each with a reference mesh of different spatial rotations.

inimetric-c3d8-ela.inp Test with four C3D8 elements, each with a reference mesh of different spatial rotations.

inimetric-c3d8i-ela.inp Test with four C3D8I elements, each with a reference mesh of different spatial rotations.

Figures

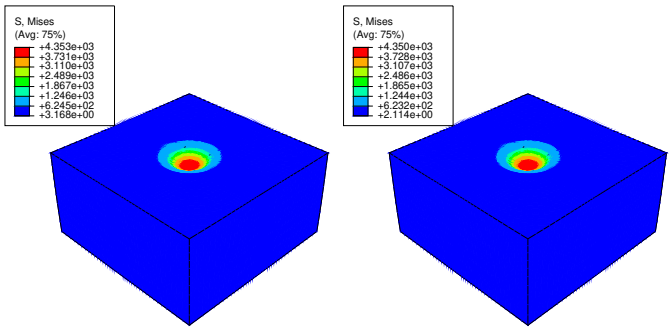


Figure 1: Indentation test: Verification for reference mesh method.

Bending stabilization for full-integration linear elements

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D8H

Features tested

Bending stabilization for full-integration linear elements modeled with hyperelastic materials is verified.

Problem description

Each model consists of three elements that are subjected to the same imposed displacement field. The constitutive responses for all three elements are assumed to be hyperelastic with identical material properties. However, they differ in the manner in which the material model is defined. One of the elements uses a built-in hyperelastic response. The other two elements use the same hyperelastic response but defined using user subroutines *UHYPER* and *UMAT*, respectively. The imposed displacement field results in a nonhomogenous deformation field within the element.

Three separate models are tested with the above basic setup. In the first model, the bending stabilization is specified to be active. In the second model, the bending stabilization is specified to be inactive. In the third model, the default behavior for the bending stabilization is used

Model: The model consists of three C3D8H elements, each of dimensions $1 \times 1 \times 1$.

Material:

Hyperelasticity: Mooney-Rivlin

$C_{10} = 80.0$

$C_{01} = 0.0$

$D_1 = 2.013\text{E}{-4}$

Boundary conditions: Each element type is loaded in displacement control to generate a nonhomogeneous deformation field.

Results and discussion

When bending stabilization is specified to be active, the elements that use user-defined materials yield the same nodal reaction forces as the element that uses the built-in material. The nodal reaction forces change identically for all three elements when the bending stabilization is specified to be inactive. For the default case, the nodal forces for the elements with the built-in hyperelastic material are identical to the case with active bending

stabilization, while the nodal forces for the elements that use user-defined materials are identical to the case with inactive bending stabilization.

Input files

umatuhyper_c3d8h_bsON.inp	Model for active bending stabilization case.
umatuhyper_c3d8h_bsOFF.inp	Model for inactive bending stabilization case.
umatuhyper_c3d8h_defaultbs.inp	Model for default bending stabilization case.
umatuhyper.f	User subroutines used in above three cases.

Interactions Verification

This section defines the basic tests that are used to provide evidence that the implementation of the interaction options in Abaqus produces the expected results.

In this section:

- *Small engineering tests with contact*
- *Contact properties*
- *Interference fit and specified clearance*
- *Contact formulations*
- *Surface smoothing for surface-to-surface contact*
- *Initial contact stress*
- *Contact wear modeling*
- *Contact surface representation of threads*
- *Miscellanous*

Small engineering tests with contact

This section defines the basic tests that are used to provide evidence that the implementation of the constraint options in Abaqus produces the expected results.

In this section:

- *Rolling of steel plate*
- *Beam impact on cylinder*

Rolling of steel plate

Products: Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

CPE4R R2D2 C3D8R R3D4

Features tested

Large deformation kinematics, user material, kinematic contact, penalty contact, friction, analytical rigid surfaces, multiple steps, adiabatic heat generation, adding contact surfaces and boundary conditions after the first step.

Problem description

This verification problem is similar to the problem described in *Rolling of thick plates*. Here, a two-dimensional, plane strain case of the rolling problem is considered with a much coarser mesh for the steel plate. The plate is modeled using plane strain elements (CPE4R) and 8-node brick elements (C3D8R). In the three-dimensional model all out-of-plane degrees of freedom are prescribed as zero to represent a state of plane strain.

The steel plate has a total thickness of 40 mm and a length of 100 mm. This analysis simulates the rolling of the plate through two roller stands, each of which achieves a reduction in the thickness of the plate of 10 mm. The radius of each roller is 50 mm. The model takes advantage of half-symmetry.

The material is modeled as an elastic, perfectly plastic material with Young's modulus 210 GPa, Poisson's ratio 0.30, yield stress 250 MPa, and density 7500 kg/m^3. The two-dimensional case uses user-defined material behavior, along with user subroutine *VUMAT*. This model can be selected by specifying the material name ABQTEST1. The user subroutine has the option to include kinematic hardening. However, this example problem tests the user material only for the case of perfect plasticity and verifies the results by comparison with the results obtained with the standard plasticity model with no hardening (for the three-dimensional case). The rotating cylinder problem of *VUMAT* verifies the hardening case for the user material. The three-dimensional model uses the standard elastic, perfectly plastic material model specified with elastic and plastic material properties. It also tests the adiabatic heat generation capability, specific heat, thermal expansion material property, and inelastic heat fraction. The initial temperature for all nodes in the model is 294°C. The specific heat for this material is 460.46 joule/kg/°C.

The rolling process is analyzed in two steps. In the first step only the first roller has a prescribed rotational velocity. The second step begins just as the plate is about to reach the second roller. At this time a prescribed velocity boundary condition is added that determines the rotational velocity of the second roller. The coefficient of friction between the rollers and the plate is 0.3. The maximum traction due to friction is assumed to be $\sigma_0/\sqrt{3}$, or 144.3 MPa.

The contact constraints can be enforced either kinematically or with a penalty method in Abaqus/Explicit. Kinematic contact gives strict enforcement of the constraints, whereas penalty contact will allow some penetration. However, the two constraint methods will usually give nearly the same results for problems that involve plastic deformation (such as rolling problems), because the contact penetrations with penalty contact will tend to be small. This is related to the fact that the default penalty stiffness is about 10% of the elastic stiffness in the elements along the contact interface. When the material yields, the penalty stiffness will typically be much larger

than the effective stiffness of the material, so the penetrations will be rather insignificant. For problems in which the material remains elastic (see *The Hertz contact problem*), the contact penetrations can be significant if the penalty method is used. While kinematic contact is available only with the contact pair capability, penalty contact is available with both the contact pair capability and the general contact capability in Abaqus/Explicit. For this analysis all three approaches to enforcing the contact constraints are used: kinematic contact with contact pair, penalty contact with contact pair, and general contact. In the first step of the analyses involving contact pairs, when only the first roller has a prescribed rotational velocity, only one contact pair is defined. This contact pair contains the surface of the first roller and the outer surface of the plate. At the start of the second step, when the plate is just about to reach the second roller, a second contact pair is introduced that contains the surface of the second roller and the outer surface of the plate. For the analysis using general contact, the default internally generated all-inclusive contact surface is referenced using the general contact inclusions procedure; hence, the contact definitions do not need to be modified from step to step.

The roller speed used for both rollers in this example is 600 rad/s. See *Rolling of thick plates* for a detailed discussion of the choice of rolling speeds.

Results and discussion

Figure 1 shows the original mesh for the two-dimensional model. *Figure 2* shows contours of shear stress at the end of the first step for the two-dimensional model. Note that the first roller has rotated during the first step, whereas the second roller remains motionless. *Figure 3* shows contours of shear stress at the end of the second step for the two-dimensional model. *Figure 4* shows contours of equivalent plastic strain (SDV5) at the end of the second step for the two-dimensional model. Since the user subroutine stores the values of equivalent plastic strain as the fifth state variable, contour plots are generated by the use of the variable SDV5.

Figure 5 contains a wire frame drawing of the original mesh for the three-dimensional model. *Figure 6* shows contours of shear stress at the end of the first step for the three-dimensional model. *Figure 7* shows contours of shear stress at the end of the second step for the three-dimensional model. *Figure 8* shows contours of equivalent plastic strain (PEEQ) at the end of the second step for the three-dimensional model. *Figure 9* shows contours of temperature at the end of the second step for the three-dimensional model. Note that the prescription of adiabatic heat generation in this example does not have an effect on the overall solution because none of the material properties are temperature dependent. It is simply used to calculate the temperature field obtained from the dissipated plastic work.

Input files

roll2dapa_anl.inp	Two-dimensional kinematic contact analysis using an analytical rigid surface.
roll3dapa_rev_anl.inp	Three-dimensional kinematic contact analysis using an analytical rigid surface of TYPE=REVOLUTION.
roll3dapa_rev_anl_gcont.inp	Three-dimensional general contact analysis using an analytical rigid surface of TYPE=REVOLUTION.
roll2dapa.inp	Two-dimensional kinematic contact analysis using rigid elements.
roll3dapa.inp	Three-dimensional kinematic contact analysis using rigid elements.
roll3dapa_gcont.inp	Three-dimensional general contact analysis using rigid elements.
roll3dapa_cyl_anl.inp	Three-dimensional kinematic contact analysis using an analytical rigid surface of TYPE=CYLINDER.
roll3dapa_cyl_anl_gcont.inp	Three-dimensional general contact analysis using an analytical rigid surface of TYPE=CYLINDER.
roll2dapa_anl_pnlty.inp	Two-dimensional penalty contact analysis using an analytical rigid surface.
roll3dapa_rev_pnlty.inp	Three-dimensional penalty contact analysis using an analytical rigid surface of TYPE=REVOLUTION.

Figures

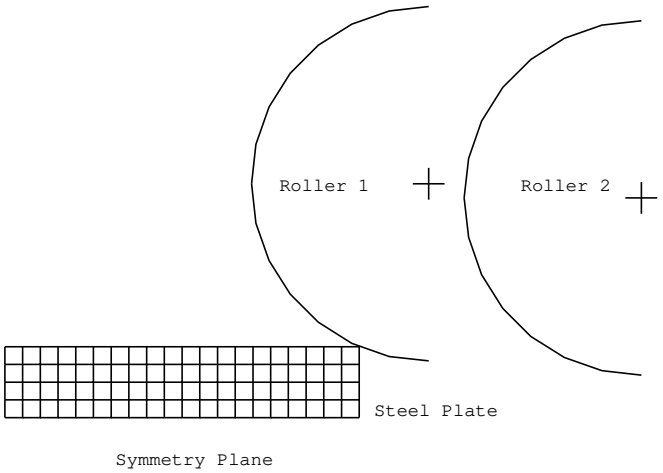


Figure 1: Undeformed mesh for the two-dimensional model.

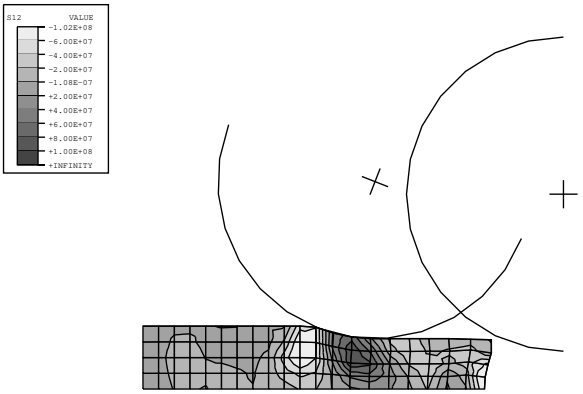


Figure 2: Contours of shear stress at the end of Step 1 for the two-dimensional model.

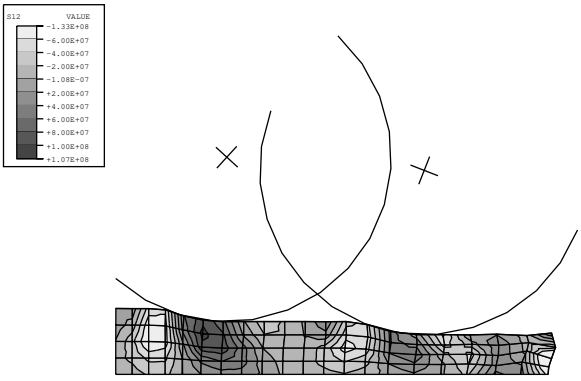


Figure 3: Contours of shear stress at the end of Step 2 for the two-dimensional model.

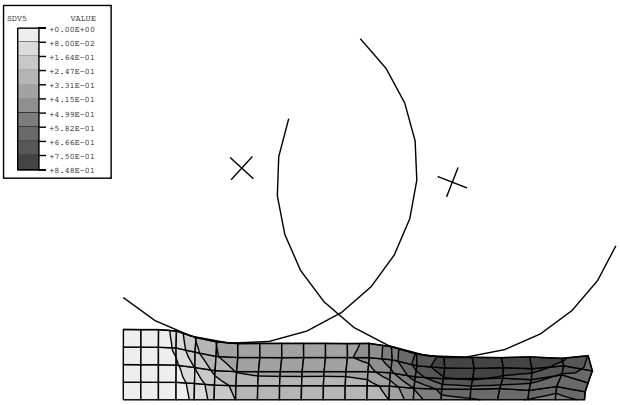


Figure 4: Contours of equivalent plastic strain at the end of Step 2 for the two-dimensional model.

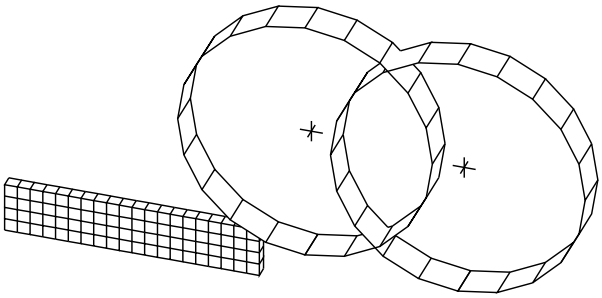


Figure 5: Undeformed mesh for the three-dimensional model.

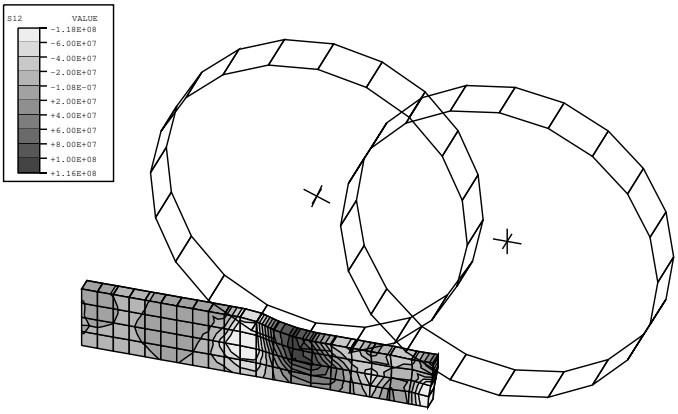


Figure 6: Contours of shear stress at the end of Step 1 for the three-dimensional model.

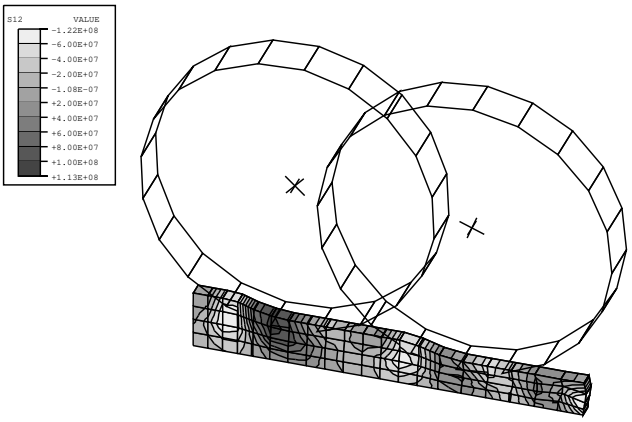


Figure 7: Contours of shear stress at the end of Step 2 for the three-dimensional model.

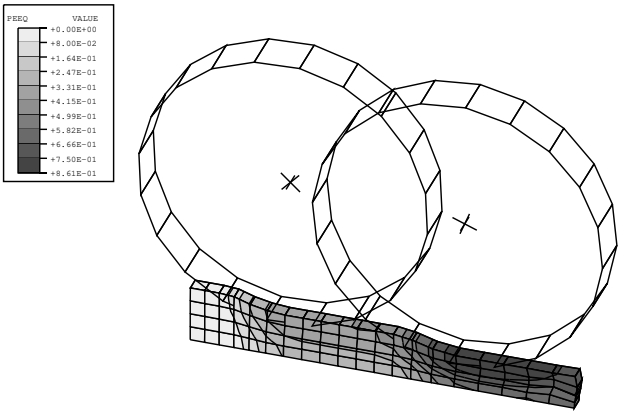


Figure 8: Contours of equivalent plastic strain at the end of Step 2 for the three-dimensional model.

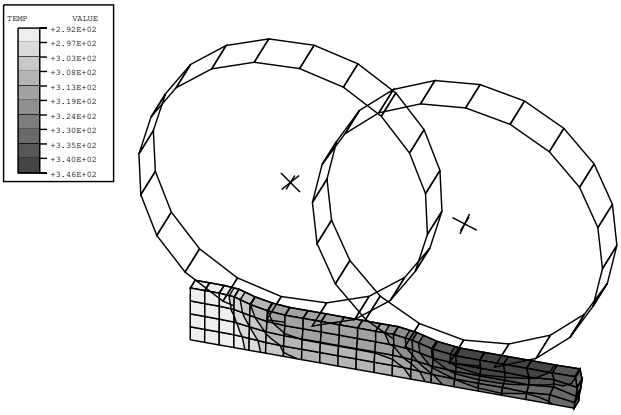


Figure 9: Contours of temperature at the end of Step 2 for the three-dimensional model.

Beam impact on cylinder

Products: Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

S4R R3D4

Features tested

Distributed loads, kinematic contact, penalty contact, analytical rigid surfaces, rigid bodies.

Problem description

This problem involves the analysis of the dynamic response of a cantilever beam subjected to a sudden, impulsively applied, pressure loading. Two cases are considered. First, the response of the cantilever beam is determined. In this case the beam responds in the first bending mode. In the second case a rigid cylinder is introduced beneath the beam and the beam strikes it.

The beam is 500 mm long and 100 mm wide and has a thickness of 2.5 mm. Half of the beam is modeled with a 20×3 mesh of shell elements using symmetry boundary conditions along the centerline of the beam. The beam is made of steel, with a Young's modulus of 200 GPa and a Poisson's ratio of 0.3. The density is 7800 kg/m^3. A von Mises elastic, perfectly plastic material model is used with a yield stress of 250 MPa.

The beam is subjected to a constant downward pressure of 0.1 MPa applied instantaneously at the beginning of the step, as shown in *Figure 1*.

In the second case a fixed, rigid cylinder of radius 40 mm is introduced, as shown in *Figure 2*. Contact surfaces are defined on the lower surface of the beam and the outer surface of the cylinder. Tests are conducted with both kinematic enforcement and penalty enforcement of the contact constraints. Kinematic contact is the default, but the penalty method can be specified as an alternative.

Two approaches for modeling the cylindrical surface are tested: using rigid elements and using analytical rigid surfaces. Analytical rigid surfaces are typically the preferred means for representing simple rigid geometries such as this in terms of both accuracy and computational performance. However, analytical surfaces always act as a pure main surface, and penetrations of a main surface into regions between secondary nodes can occur without generating contact forces (see *Contact Constraint Enforcement Methods in Abaqus/Explicit*). These penetrations may be significant if the secondary surface is coarsely discretized. In these cases it may be preferable to use an element-based rigid surface and balanced main-secondary penalty contact. Weighting of a rigid surface as a secondary surface is allowed only if it is element-based (not an analytical surface) and penalty contact is used.

Additional refinement of the rigid surface in the cylindrical direction has been used for the model in which the rigid surface nodes act partially as secondary nodes so that penetrations of the rigid surface into the deformable surface are detected. This refinement adds some computational cost, but it does not affect the stable time increment. Cylindrical refinement would not influence the contact compliance when the rigid surface acts as a pure main surface, so this type of refinement is not used in these cases.

A further comment on rigid surface modeling is that complex three-dimensional surface geometries that often occur in practice must be modeled with element-based rigid surfaces.

Results and discussion

Verification for this problem is provided by comparing the values of significant problem variables with the values produced by an equivalent model in Abaqus/Standard. The Abaqus/Standard analyses use 5-point Simpson integration only and a half-increment residual force tolerance value of 1.0×10^3. The Abaqus/Explicit analyses are run with 5-point Simpson integration and 3-point Gauss integration. The rigid surface is modeled as analytical and acts as a pure main surface in the Abaqus/Standard analysis. The contact constraints account for the shell thickness in the Abaqus/Explicit analyses only. The Abaqus/Explicit results shown below are for an element-based rigid surface with kinematic enforcement of contact constraints, except where noted otherwise.

Table 1 and *Table 2* compare tip displacements, tip velocities, and whole model energies at several points along the beam's symmetry axis. Tip displacements and velocities are averaged over the four nodes at the tip of the beam. The results from the Abaqus/Explicit analyses using Simpson (5-point) and Gauss (3-point) integration through the thickness of the shell demonstrate slight sensitivity of the response to the choice of the integration rule. Corresponding components of displacement and velocity at the tip of the beam are within 0.1% and 0.5%, respectively, for the Abaqus/Explicit (Simpson integration) and Abaqus/Standard analyses without the cylinder. For the problem with the cylinder, the significant components of displacement and velocity are within 2% and 8%, respectively, between the Abaqus/Explicit and Abaqus/Standard results with Simpson integration.

Figure 3 shows contours of equivalent plastic strain on the bottom surface of the beam for the Abaqus/Explicit analysis using Simpson integration without the rigid cylinder. *Figure 4* shows the corresponding plot for the Abaqus/Standard analysis. The contours are plotted on the deformed shapes of the beam. After 0.08 seconds a plastic hinge has formed at the fixed end of the beam for both cases.

Figure 5 and *Figure 6* show contours of equivalent plastic strain on the bottom surface of the beam impacting the rigid cylinder for the Abaqus/Explicit analysis with Simpson integration and the Abaqus/Standard analysis, respectively.

Figure 7 through *Figure 10* show the final configuration near the rigid cylinder for four Abaqus/Explicit analyses. *Figure 7* corresponds to an analysis with an analytical rigid surface and kinematic contact. *Figure 8* corresponds to an analysis with an analytical rigid surface and penalty contact. In both of these cases the analytical surface is the pure main surface of the contact pair. Contact is enforced at the secondary nodes accounting for the shell thickness, and there is some penetration of the rigid surface into the shell. The final position of the tip is slightly different in *Figure 7* and *Figure 8*, which is attributable to impacts being perfectly plastic with kinematic contact and elastic with penalty contact (see *Contact Constraint Enforcement Methods in Abaqus/Explicit*). *Figure 9* corresponds to an analysis with an element-based rigid surface and kinematic contact. *Figure 10* corresponds to an analysis with an element-based rigid surface and penalty contact. Penetration of the rigid surface into the shell surface is repelled only in *Figure 10* because this is the only case in which the rigid surface nodes are weighted at all as secondary nodes.

Input files

beamimpac1.inp	Simpson integration case without the rigid cylinder.
beamimpac2.inp	Simpson integration case with the rigid cylinder.
beamimpac2_cyl_anl.inp	Explicit dynamic analysis using an analytical rigid surface.
beamimpac2_rev_anl.inp	Explicit dynamic analysis using an analytical rigid surface.
beamimpac2_pnlty.inp	Explicit dynamic analysis using an element-based rigid surface and penalty contact.
beamimpac2_gcont.inp	Explicit dynamic analysis using an element-based rigid surface and the general contact capability.
beamimpac2_gcont_subcyc.inp	Explicit dynamic analysis using an element-based rigid surface and the general contact capability and subcycling.

beamimpac2_rev_pnlty.inp	Explicit dynamic analysis using an analytical rigid surface and penalty contact.
beamimpac1_gauss.inp	Gauss integration explicit dynamic analysis of the case without the rigid cylinder.
beamimpac2_gauss.inp	Gauss integration explicit dynamic analysis of the case with the rigid cylinder.
beamstandard1.inp	Implicit dynamic analysis of the case without the rigid cylinder.
beamstandard2.inp	Implicit dynamic analysis of the case with the rigid cylinder using the hard contact model.
beamstandard2_auglagr.inp	Implicit dynamic analysis of the case with the rigid cylinder using the augmented Lagrangian contact model.
beamimpac2_offset.inp	Explicit dynamic analysis of the case with the rigid cylinder that demonstrates the effects of shell offset and rigid thickness on contact surfaces.

Tables

Table 1: Comparison of results for case without rigid cylinder (results obtained on an SGI R4600 using single precision).

Variable	Abaqus/Explicit		Abaqus/Standard
	(Gauss)	(Simpson)	(Simpson)
u_x^{tip} (mm)	−115	−114	−114
u_z^{tip} (mm)	−292	−293	−293
v_x^{tip} (m/s)	−45.4	−45.6	−45.5
v_z^{tip} (m/s)	−64.7	−65.0	−64.8
ALLKE (joules)	434	429	428
ALLIE (joules)	29.3	31.7	31.6
ETOTAL (joules)	7.3×10^{-2}	6.95×10^{-2}	−1.58

Table 2: Comparison of results for case with rigid cylinder (results obtained on an SGI R4600 using single precision).

Variable	Abaqus/Explicit		Abaqus/Standard
	(Gauss)	(Simpson)	(Simpson)
u_x^{tip} (mm)	−253	−248	−250
u_z^{tip} (mm)	−122	−141	−143
v_x^{tip} (m/s)	−20.8	−38.0	−41.0
v_z^{tip} (m/s)	−77.2	−56.0	−56.9
ALLKE (joules)	82.0	83.8	86.8

Variable	Abaqus/Explicit		Abaqus/Standard
	(Gauss)	(Simpson)	(Simpson)
ALLIE (joules)	114	112	112
ETOTAL (joules)	0.528	0.380	–0.654

Figures

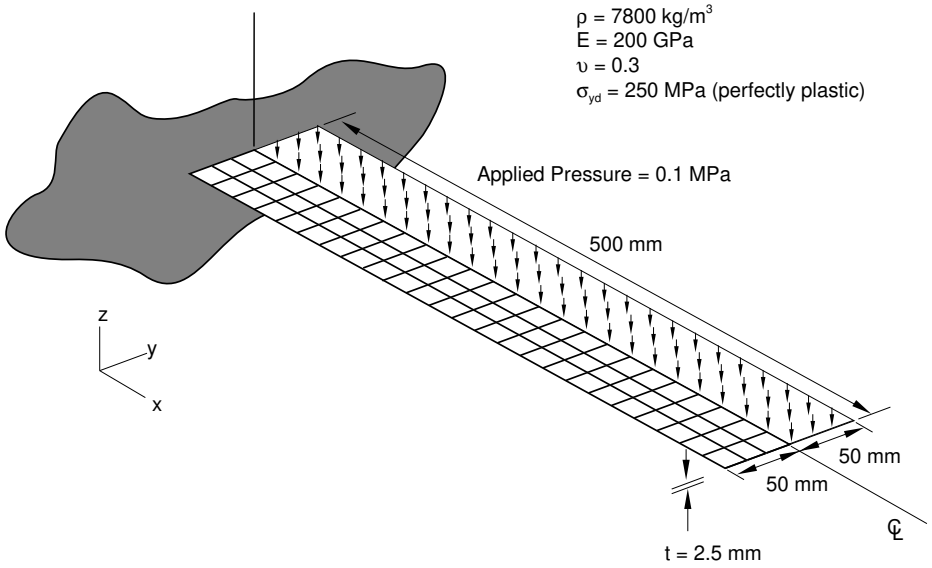


Figure 1: Impulsively loaded cantilever beam.

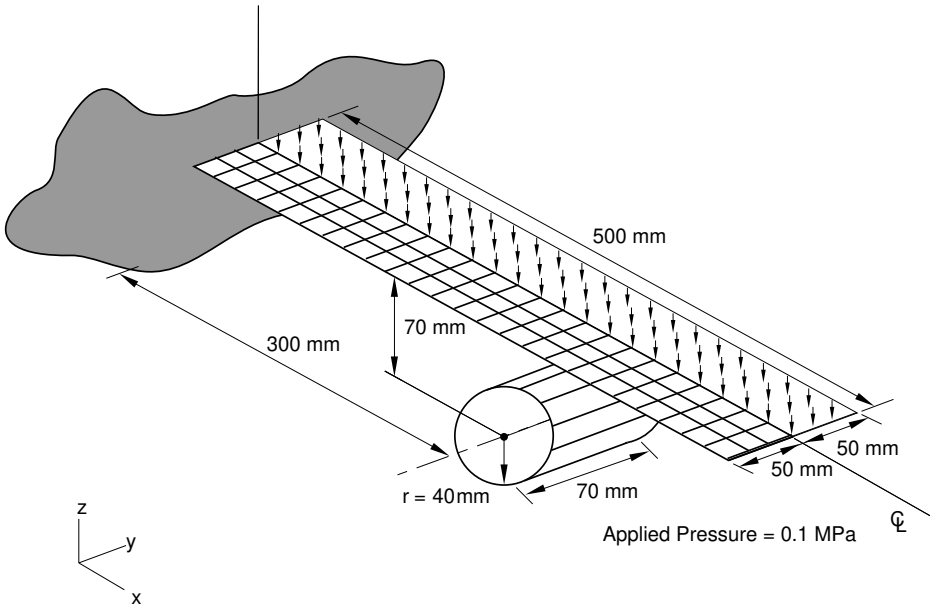


Figure 2: Cantilever beam impacting on a rigid cylinder.

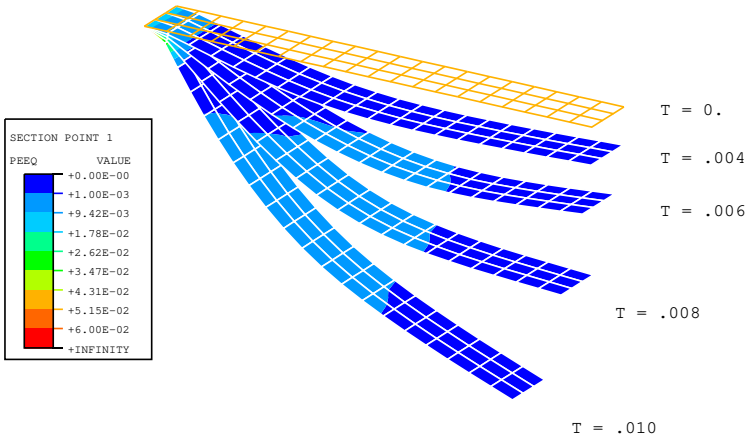


Figure 3: Plastic strain on bottom surface of beam, Abaqus/Explicit analysis.

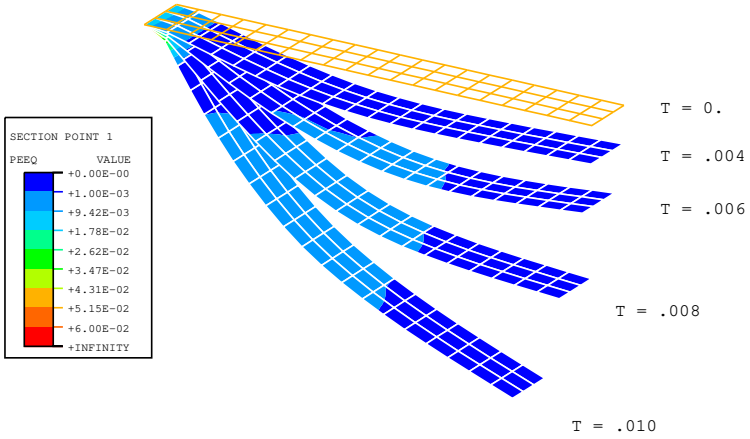


Figure 4: Plastic strain on bottom surface of beam, Abaqus/Standard analysis.

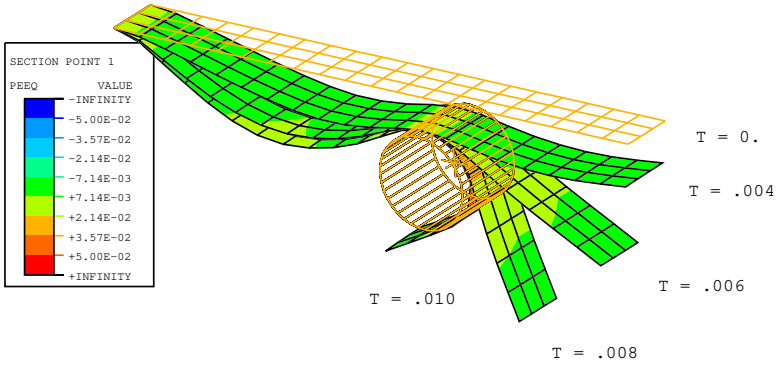


Figure 5: Plastic strain on bottom surface of beam, Abaqus/Explicit analysis.

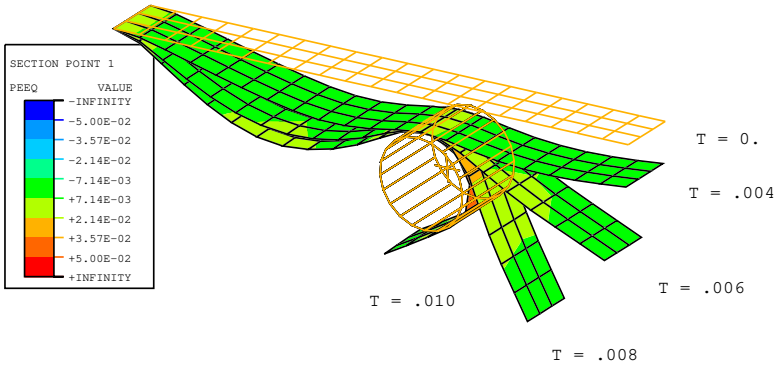


Figure 6: Plastic strain on bottom surface of beam, Abaqus/Standard analysis.

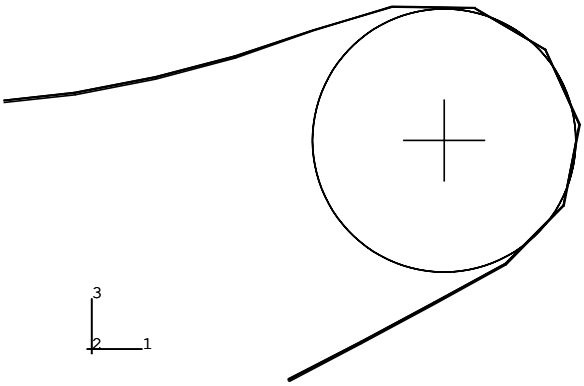


Figure 7: Deformed configuration near rigid cylinder for an analytical rigid surface and kinematic contact.

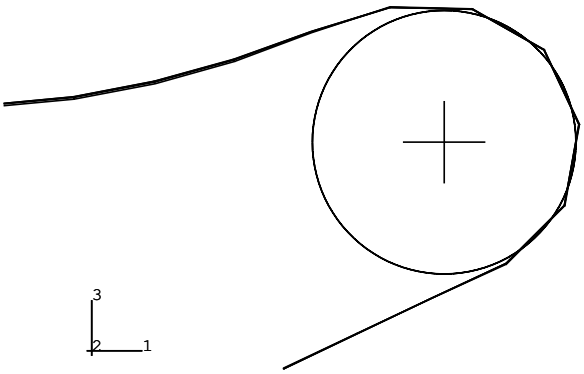


Figure 8: Deformed configuration near rigid cylinder for an analytical rigid surface and penalty contact.

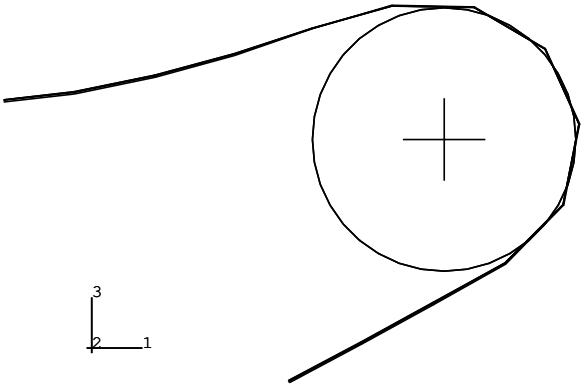


Figure 9: Deformed configuration near rigid cylinder for an element-based rigid surface and kinematic contact.

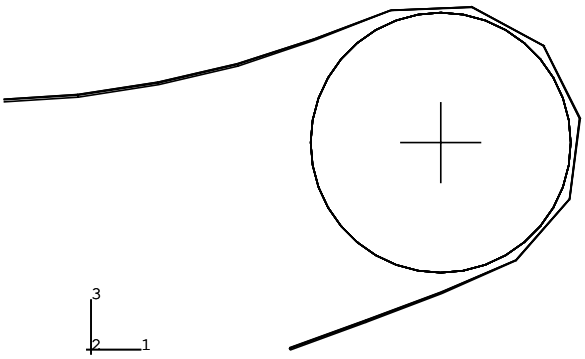


Figure 10: Deformed configuration near rigid cylinder for an element-based rigid surface and penalty contact.

Contact properties

This section defines the basic tests that are used to provide evidence that the implementation of the constraint options in Abaqus produces the expected results.

In this section:

- *Anisotropic friction for Abaqus/Explicit*
- *Thermal surface interaction*
- *Coupled thermal-electrical surface interaction*
- *Friction models in Abaqus/Standard*
- *Friction models in Abaqus/Explicit*
- *Cohesive surface interaction*
- *Modified contact pressure-overclosure relationship*

Anisotropic friction for Abaqus/Explicit

Products: Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D8R S4R

Features tested

Anisotropic friction for general contact in Abaqus/Explicit.

Problem description

The models consist of a solid element pressed against a shell element, converted to a rigid element for convenience, and displaced along the shell surface in varying directions. At the beginning of the analysis both contact surfaces have a directional preference oriented with the major axis of the scaling ellipse pointing in the *Y*-direction (see the vector plot of CORIENT1 in *Figure 1*). The frictional directional preference factor, ϵ, for both surfaces is equal to 0.5. The nominal or average friction coefficient, μ_{iso}, is 0.2.

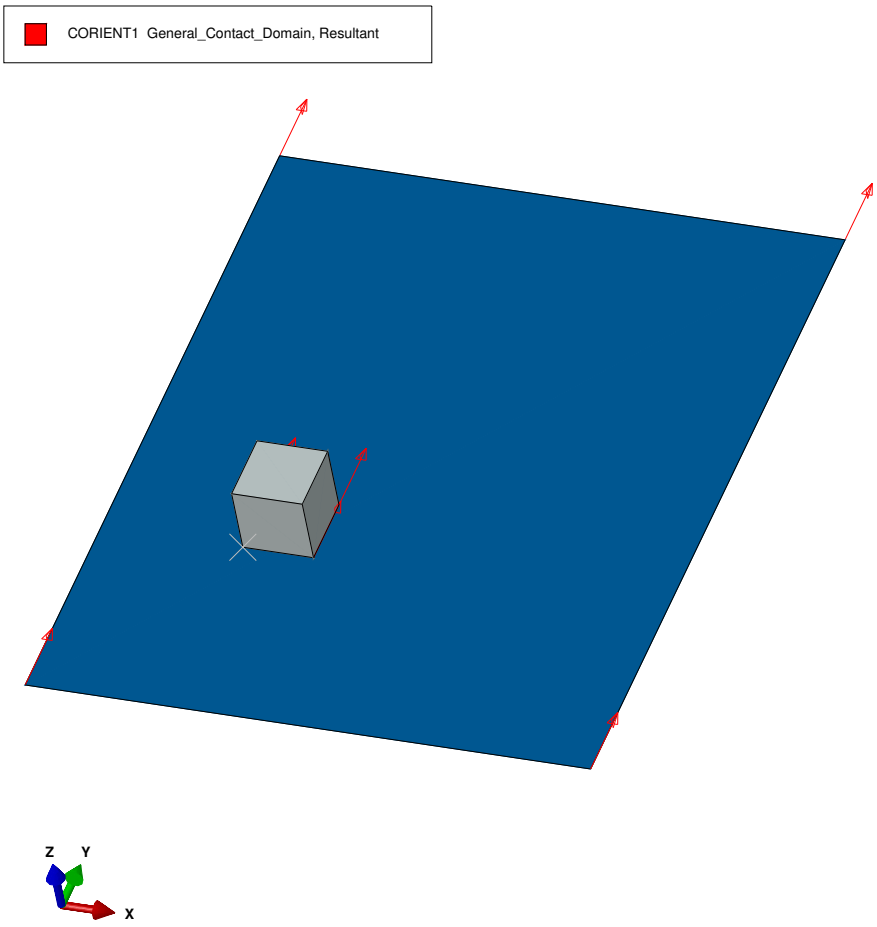


Figure 1: Vector output CORIENT1.

Model: Model-1:

Step-1	The block is pressed on the shell element with a force of 25 N.
Step-2	The block is displaced with a boundary condition in the *X*-direction.
Step-3	Release displacement in the *X*-direction.
Step-4	The block is displaced with a boundary condition in the *Y*-direction.
Step-5	Release displacement in *Y*-direction.
Step-6	The block is displaced with a boundary condition in the *X*–*Y* direction (45°).

Model-2:

Step-1	The block is pressed on the shell element with a force of 25 N.
Step-2	The block is displaced with a boundary condition in the *X*-direction.
Step-3	Release displacement in *X*-direction and pressure.
Step-4	Rotate the shell element with a rotation of 90°.

Step-5	Apply the pressure again.
Step-6	The block is displaced again with a boundary condition in the *X*-direction.

After the shell is rotated, the two surfaces have orthogonal directional preferences and the equivalent scaling ellipse becomes a circle.

Material:

Density	$7.8\,e^{-9}$
Young's modulus	$7.0\,e^{4}$
Poisson's ratio	0.3

Anisotropic behavior:: The scaling ellipses can be mathematically treated as tensors. The form in the principal coordinate system is

$$S = \begin{bmatrix} 1+\epsilon & 0 \\ 0 & 1-\epsilon \end{bmatrix}.$$

The form in a rotated coordinate system is

$$S = RSR^T$$

$$R = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix}$$

$$S = R\begin{bmatrix} 1+\epsilon & 0 \\ 0 & 1-\epsilon \end{bmatrix}R^T$$

$$S = \begin{bmatrix} 1+\epsilon\cos(2\theta) & -\epsilon\sin(2\theta) \\ -\epsilon\sin(2\theta) & 1-\epsilon\cos(2\theta) \end{bmatrix}.$$

Each ellipse is defined as

$$S_{surf1} = S_{surf2} = \begin{bmatrix} 1+0.5 & 0 \\ 0 & 1-0.5 \end{bmatrix} = \begin{bmatrix} 1.5 & 0 \\ 0 & 0.5 \end{bmatrix}.$$

The two surface scaling tensors (ellipses) are averaged to determine the contact scaling tensor (ellipse):

$$S_{cont} = \omega_1 S_{surf1} + \omega_2 S_{surf2}.$$

Initially the major axis of the two ellipses are aligned, and using a balanced weighting method, the contact scaling ellipse becomes

$$S_{cont} = 0.5\begin{bmatrix} 1.5 & 0 \\ 0 & 0.5 \end{bmatrix} + 0.5\begin{bmatrix} 1.5 & 0 \\ 0 & 0.5 \end{bmatrix} = \begin{bmatrix} 1.5 & 0 \\ 0 & 0.5 \end{bmatrix}.$$

The limit shear force in the principal direction is computed by multiplying S by the average friction coefficient and the normal force,

$$F_{Limit} = F_N\, S_{cont}\, \mu_{iso} = 25\begin{bmatrix} 1.5 & 0 \\ 0 & 0.5 \end{bmatrix}0.2 = \begin{bmatrix} 7.5 & 0 \\ 0 & 2.5 \end{bmatrix}.$$

Results and discussion

The reaction forces are checked and compared with the analytical results.

- Model-1: since the equivalent scaling ellipse is oriented as the *Y*-direction and the two surfaces are not rotating elatively, the expected reaction forces are equal to Fx = 2.5 N (Step-2) and Fy = 7.5 N (Step-4) during the sliding in the *X*- and *Y*-directions.(see *Figure 3*). In Step-6 the block is sliding in a diagonal direction in the X–Y plane (45°). By definition, the direction of the slip is orthogonal to the equivalent scaling ellipse (see *Figure 2*).

  $$F_X = Ak_x y_x^{el},\ F_y = Ak_y y_y^{el},$$

  where

  $$\frac{k_y}{k_x} = \left(\frac{1+\varepsilon}{1-\varepsilon}\right)^2 = \left(\frac{1.5}{0.5}\right)^2 = 9.$$

  For $y_x^{el} = y_y^{el}$ (45° sliding),

  $$\frac{F_y}{F_x} = 9, \theta = \tan^{-1}(9) \approx 83.6^\circ.$$

  The values of the forces F_x and F_y computed intersecting the scaling ellipse using the angle $\theta = 83.6^\circ$ are

  $F_x = 25 * 0.033 = 0.825$ and $F_y = 25 * 0.284 = 7.1$ (see *Figure 3*).
- Model-2: during Step-2 the equivalent scaling ellipse is oriented as the *Y*-direction, so the expected reaction force is Fx = 2.5 N. After the rotation of the shell, the directional preferences of the two surfaces become orthogonal, generating an equivalent scaling ellipse equal to a circle for the isotropic case (see *Figure 4*).

Input files

xpl_anisFric_1.inp	Input file for Model-1.
xpl_anisFric_2.inp	Input file for Model-2.

Figures

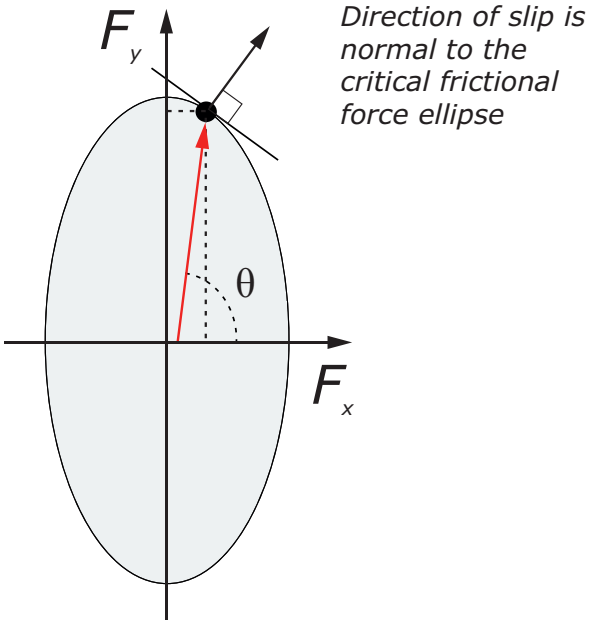


Figure 2: Direction of slip respect to the critical force ellipse.

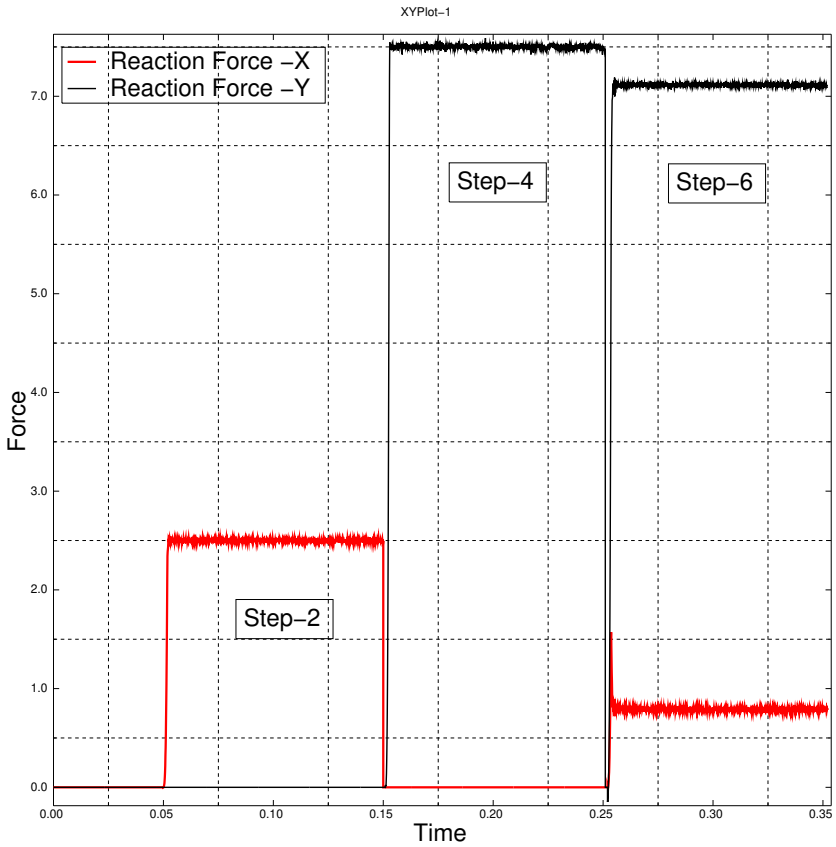


Figure 3: Reaction forces for Model-1.

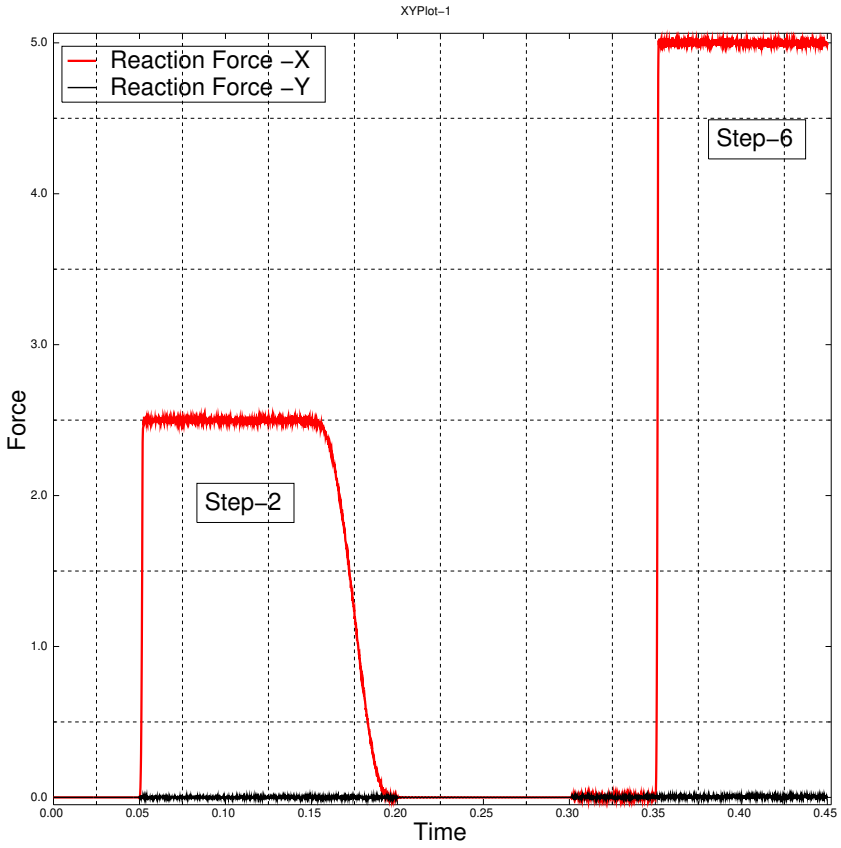


Figure 4: Reaction forces for Model-2.

Thermal surface interaction

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

DC1D2 DC2D4 DC2D8 DC3D8 DC3D20 DCAX4 DCAX8
DS3 DS4 DS6 DS8 DSAX1 DSAX2
CAX3T CAX4RT CPE3T CPE4RT CPS3T CPS4RT
C3D4T C3D6T C3D8RT SC8RT SC6RT

Features tested

Contact pair

Gap conductance

Gap radiation

Problem description

A solid material is placed near a heat source whose temperature stays constant. Heat transfer across the gap between the solid surface and the heat source can take place via gap conductance or gap radiation (thus, there are two tests for each element type). Using the default convergence tolerances in Abaqus/Standard, the reaction fluxes for radiation problems show slight differences (0.1%) from the analytical results due to the severe nonlinearity of the radiation problem. We initiate unidirectional heat flow by applying a constant temperature that is higher than that of the heat source itself to the solid surface away from the heat source. The steady-state temperature at the surface near the heat source is used to verify the numerical solutions.

In Abaqus/Explicit the steady-state result is obtained by performing a long-term transient simulation. The constant temperature heat source is modeled three different ways: with either deformable elements, isothermal discrete rigid elements, or an isothermal analytical rigid surface.

Model:

Element size	1 unit in each direction
Inner radius of axisymmetric solids	10.0

Material:

Conductivity in solid	1.0
Gap conductance	10.0
Radiation constants of surfaces	5×10^{-10}
Absolute zero	−460.0

In Abaqus/Explicit dummy mechanical and capacitance properties are specified to complete the material definition.

Results and discussion

The steady-state temperatures agree with the analytical, one-dimensional heat transfer results.

Input files

Abaqus/Standard input files

Gap conductance tests:

eiu1dgccc.inp	DC1D2, DGAP elements.
ei22discc.inp	DC2D4 elements.
ei23discc.inp	DC2D8 elements.
ei23discc_gc.inp	DC2D8 elements with general contact.
ei34discc.inp	DC3D8 elements.
ei38discc.inp	DC3D20 elements.
ei38discc_gc.inp	DC3D20 elements with general contact.
eia2discc.inp	DCAX4 elements.
eia2discc_gc.inp	DCAX4 elements with general contact.
eia3discc.inp	DCAX8 elements.
eia3discc_gc.inp	DCAX8 elements with general contact.
ei33discs.inp	DS3 elements.
ei33discs_gc.inp	DS3 elements with general contact.
ei33discs_interf_gc.inp	DS3 elements with deep overclosure between heat transfer surfaces modeled using general contact.
ei34discs.inp	DS4 elements.
ei34discs_gc.inp	DS4 elements with general contact.
ei34discs_interf_gc.inp	DS4 elements with deep overclosure between heat transfer surfaces modeled using general contact.
ei36discs.inp	DS6 elements.
ei36discs_gc.inp	DS6 elements with general contact.
ei36discs_interf_gc.inp	DS6 elements with deep overclosure between heat transfer surfaces modeled using general contact.
ei38discs.inp	DS8 elements.
ei38discs_gc.inp	DS8 elements with general contact.
ei38discs_interf_gc.inp	DS8 elements with deep overclosure between heat transfer surfaces modeled using general contact.
eia2discs.inp	DSAX1 elements.
eia3discs.inp	DSAX2 elements.

Gap radiation tests:

eiu1dgcrc.inp	DC1D2, DGAP elements.
ei22disrc.inp	DC2D4 elements.
ei23disrc.inp	DC2D8 elements.
ei34disrc.inp	DC3D8 elements.
ei38disrc.inp	DC3D20 elements.

eia2disrc.inp	DCAX4 elements.
eia3disrc.inp	DCAX8 elements.
ei33disrs.inp	DS3 elements.
ei33disrs_gc.inp	DS3 elements with general contact.
ei33disrs_interf_gc.inp	DS3 elements with deep overclosure between heat transfer surfaces modeled using general contact.
ei34disrs.inp	DS4 elements.
ei36disrs.inp	DS6 elements.
ei36disrs_gc.inp	DS6 elements with general contact.
ei36disrs_interf_gc.inp	DS6 elements with deep overclosure between heat transfer surfaces modeled using general contact.
ei38disrs.inp	DS8 elements.
eia2disrs.inp	DSAX1 elements.
eia2disrs_gc.inp	DSAX1 elements with general contact.
eia2disrs_interf_gc.inp	DSAX1 elements with deep overclosure between heat transfer surfaces modeled using general contact.
eia3disrs.inp	DSAX2 elements.
eia3disrs_gc.inp	DSAX2 elements with general contact.
eia3disrs_interf_gc.inp	DSAX2 elements with deep overclosure between heat transfer surfaces modeled using general contact.

Abaqus/Explicit input files

Gap conductance tests, deformable-deformable mechanical contact:

gapheattrans_c_x_df_cax3t.inp	Solid and heat source modeled with CAX3T elements.
gapheattrans_c_x_df_cax4rt.inp	Solid and heat source modeled with CAX4RT elements.
gapheattrans_c_x_df_cpe3t.inp	Solid and heat source modeled with CPE3T elements.
gapheattrans_c_x_df_cpe4rt.inp	Solid and heat source modeled with CPE4RT elements.
gapheattrans_c_x_df_cps3t.inp	Solid and heat source modeled with CPS3T elements.
gapheattrans_c_x_df_cps4rt.inp	Solid and heat source modeled with CPS4RT elements.
gapheattrans_c_x_df_c3d4t.inp	Solid and heat source modeled with C3D4T elements.
gapheattrans_c_x_df_c3d6t.inp	Solid and heat source modeled with C3D6T elements.
gapheattrans_c_x_df_c3d8rt.inp	Solid and heat source modeled with C3D8RT elements.
gapheattrans_c_x_df_sc8rt.inp	Solid and heat source modeled with SC8RT elements.

Gap conductance tests, discrete rigid-deformable mechanical contact:

gapheattrans_c_x_dr_cax4rt.inp	Solid modeled with CAX4RT elements; constant temperature heat source modeled with RAX2 elements.
gapheattrans_c_x_dr_cpe3t.inp	Solid modeled with CPE3T elements; constant temperature heat source modeled with R2D2 elements.
gapheattrans_c_x_dr_cps4rt.inp	Solid modeled with CPS4RT elements; constant temperature heat source modeled with R2D2 elements.
gapheattrans_c_x_dr_c3d4t.inp	Solid modeled with C3D4T elements; constant temperature heat source modeled with R3D3 elements.
gapheattrans_c_x_dr_c3d8rt.inp	Solid modeled with C3D8RT elements; constant temperature heat source modeled with R3D4 elements.

gapheattrans_c_x_dr_sc8rt.inp	Solid modeled with SC8RT elements; constant temperature heat source modeled with R3D4 elements.

Gap conductance tests, analytical rigid-deformable mechanical contact:

gapheattrans_c_x_ar_cax4rt.inp	Solid modeled with CAX4RT elements; constant temperature heat source modeled with an analytical rigid surface.
gapheattrans_c_x_ar_cpe3t.inp	Solid modeled with CPE3T elements; constant temperature heat source modeled with an analytical rigid surface.
gapheattrans_c_x_ar_cps4rt.inp	Solid modeled with CPS4RT elements; constant temperature heat source modeled with an analytical rigid surface.
gapheattrans_c_x_ar_c3d4t.inp	Solid modeled with C3D4T elements; constant temperature heat source modeled with an analytical rigid surface.

Gap radiation tests, deformable-deformable mechanical contact:

gapheattrans_r_x_df_cax3t.inp	Solid and heat source modeled with CAX3T elements.
gapheattrans_r_x_df_cax4rt.inp	Solid and heat source modeled with CAX4RT elements.
gapheattrans_r_x_df_cpe3t.inp	Solid and heat source modeled with CPE3T elements.
gapheattrans_r_x_df_cpe4rt.inp	Solid and heat source modeled with CPE4RT elements.
gapheattrans_r_x_df_cps3t.inp	Solid and heat source modeled with CPS3T elements.
gapheattrans_r_x_df_cps4rt.inp	Solid and heat source modeled with CPS4RT elements.
gapheattrans_r_x_df_c3d4t.inp	Solid and heat source modeled with C3D4T elements.
gapheattrans_r_x_df_c3d6t.inp	Solid and heat source modeled with C3D6T elements.
gapheattrans_r_x_df_c3d8rt.inp	Solid and heat source modeled with C3D8RT elements.
gapheattrans_r_x_df_sc8rt.inp	Solid and heat source modeled with SC8RT elements.

Gap radiation tests, discrete rigid-deformable mechanical contact:

gapheattrans_r_x_dr_cax3t.inp	Solid modeled with CAX3T elements; constant temperature heat source modeled with RAX2 elements.
gapheattrans_r_x_dr_cpe4rt.inp	Solid modeled with CPE4RT elements; constant temperature heat source modeled with R2D2 elements.
gapheattrans_r_x_dr_cps3t.inp	Solid modeled with CPS3T elements; constant temperature heat source modeled with R2D2 elements.
gapheattrans_r_x_dr_c3d6t.inp	Solid modeled with C3D6T elements; constant temperature heat source modeled with R3D4 elements.

Gap radiation tests, analytical rigid-deformable mechanical contact:

gapheattrans_r_x_ar_cax3t.inp	Solid modeled with CAX3T elements; constant temperature heat source with an analytical rigid surface.
gapheattrans_r_x_ar_cpe4rt.inp	Solid modeled with CPE4RT elements; constant temperature heat source modeled with an analytical rigid surface.
gapheattrans_r_x_ar_cps3t.inp	Solid modeled with CPS3T elements; constant temperature heat source modeled with an analytical rigid surface.
gapheattrans_r_x_ar_c3d6t.inp	Solid modeled with C3D6T elements; constant temperature heat source modeled with an analytical rigid surface.
gapheattrans_r_x_ar_c3d8rt.inp	Solid modeled with C3D8RT elements; constant temperature heat source modeled with an analytical rigid surface.
gapheattrans_r_x_ar_sc6rt.inp	Solid modeled with SC6RT elements; constant temperature heat source modeled with an analytical rigid surface.

Coupled thermal-electrical surface interaction

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

DC2D4E DC2D8E DC3D8E DC3D20E DCAX4E DCAX8E

Features tested

Contact pair

Gap conductance

Gap electrical conductance

Gap heat generation

Gap radiation

Problem description

A solid material is placed near a heat source whose temperature and electrical potential remain constant. Heat transfer across the gap between the solid body and the heat source can take place via gap conductance or gap radiation (thus, there are two tests for each element type). Electrical current is conducted between the two closely adjacent surfaces forming the gap. Half of the electrical energy resulting from this conductance is released as heat and is distributed equally to the two adjacent surfaces. No Joule heating occurs in the model as a result of electrical conduction; therefore, electrical energy does not act as an internal heat source within the continuum elements. For simplicity we initiate a unidirectional heat flow and current in the solid by applying a higher temperature and electrical potential to the face farthest from the heat source. The steady-state temperatures and electrical potentials of the solid face closest to the heat source are verified with the analytical solution.

Model:

Element size	1 unit in each direction
Inner radius of axisymmetric solids	10.0

Material:

Thermal conductivity in solid	1.0
Electrical conductivity in solid	1.0
Gap thermal conductance	10.0
Gap electrical conductance	10.0
Gap heat generation	0.5
Radiation constants of surfaces	5×10^{-10}
Joule heat fraction	0.0

Results and discussion

The steady-state temperatures and electrical potentials agree with the analytical, one-dimensional coupled thermal-electrical results.

Input files

Gap conductance tests:

ei22vsjc.inp	DC2D4E elements.
ei22vsjc_gc.inp	DC2D4E elements with general contact.
ei23vsjc.inp	DC2D8E elements.
ei23vsjc_gc.inp	DC2D8E elements with general contact.
ei34vsjc.inp	DC3D8E elements.
ei34vsjc_gc.inp	DC3D8E elements with general contact.
ei38vsjc.inp	DC3D20E elements.
ei38vsjc_gc.inp	DC3D20E elements with general contact.
eia2vsjc.inp	DCAX4E elements.
eia2vsjc_gc.inp	DCAX4E elements with general contact.
eia3vsjc.inp	DCAX8E elements.

Gap radiation tests:

ei22vsjr.inp	DC2D4E elements.
ei22vsjr_gc.inp	DC2D4E elements with general contact.
ei23vsjr.inp	DC2D8E elements.
ei23vsjr_gc.inp	DC2D8E elements with general contact.
ei34vsjr.inp	DC3D8E elements.
ei34vsjr_gc.inp	DC3D8E elements with general contact.
ei38vsjr.inp	DC3D20E elements.
ei38vsjr_gc.inp	DC3D20E elements with general contact.
eia2vsjr.inp	DCAX4E elements.
eia2vsjr_gc.inp	DCAX4E elements with general contact.
eia3vsjr.inp	DCAX8E elements.

Friction models in Abaqus/Standard

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

B21 B31

Features tested

Friction

Changes to friction properties

Problem description

The model consists of two rods perpendicular to a fixed rigid surface forced into contact with the rigid surface by a concentrated load applied in the axial direction at the top of each rod. Subsequently, shear forces are applied, such that $\tau_{eq} < \tau_{crit}$, to verify the "stick" condition. Afterward, prescribed displacements are applied to the rods to force them to slide around the surface.

The contact between the bottom end of the rod and the rigid surface is modeled by specifying a main-secondary contact pair. The bottom end of the rod constitutes the secondary surface created with the node-based surface procedure and has a contact area of unity; hence, the normal force applied on the rod is equal to the contact pressure. Each rod has its separate surface interaction created with the contact property and friction definitions. Further, friction properties are modified during the analysis.

Model:

Average length of all contact elements	0.5

Material:

Young's modulus	30×10^6
Poisson's ratio	0.3

Coulomb friction model

The first two steps of the analysis establish contact between each rod and the rigid surface and set up an equilibrium solution in which each beam element is compressed by a force of 300. The temperature of the secondary node is specified as 20° and that of the rigid surface, as 0°; therefore, the average surface temperature is 10° when contact is established. In Step 3 the normal force is increased to 400, and a shear force is applied to the first rod such that $\tau_{eq} < \tau_{crit}$ and the rod remains sticking. The shear force is removed in Step 4. In Step 5 the friction model for rod 1 is modified. The normal force is increased to 550, and a shear force is applied such that $\tau_{eq} < \tau_{crit}$ and the rod still remains sticking. The shear forces are removed in Step 6. In Step 7 the original friction model

is specified with the friction properties reset to their original values. The pressure on rod 1 is increased to 850, and a slip is applied. In Step 8 a slip velocity–dependent friction model is introduced for rod 2. In Step 9 a slip is applied to rod 2 in which the slip rate is varied by prescribing the displacement with an amplitude curve during the static step.

Surface interaction for rod 1:

Step 1

$\mu_1 = 0.005\overline{\theta} + 2.5 \times 10^{-4}(p - 100)$.

$\mu_2 = 0.005\overline{\theta} + 3.3 \times 10^{-4}(p - 100)$. (Anisotropic model only for case with 2 local tangent directions.)

Step 5

$\mu = 0.002\overline{\theta} + 3.3 \times 10^{-4} p$ for $100 < p \leq 500$.

$\mu = 0.1650 + 0.002\overline{\theta} + 5.5 \times 10^{-4}(p - 500)$ for $500 < p \leq 900$.

Step 7

Same as specified in Step 1.

Step 8

$\mu = 0.0$ for the elastic slip formulation.

Rough friction model for the Lagrange multiplier formulation.

Surface interaction for rod 2:

Step 1

$\mu = 0.0$ for the elastic slip formulation.

Rough friction model for the Lagrange multiplier formulation.

Step 8

$\mu = \mu_s$ for $\dot{\gamma}_{eq} = 0$;

$\mu = \mu_s - (\mu_s - \mu_k)\,\dot{\gamma}_{eq}/2.0$ for $0 < \dot{\gamma}_{eq} \leq 2.0$;

$\mu = \mu_k$ for $\dot{\gamma}_{eq} > 2.0$, where

$\mu_s = 0.2$ and $\mu_k = 0.0$ for $p = 100.0$ and

$\mu_s = 0.4$ and $\mu_k = 0.2$ for $p = 500.0$.

Exponential decay model

The first two steps of the analysis establish contact between each rod and the rigid surface and set up an equilibrium solution in which each beam element is compressed by a force of 300. The pressure is kept constant throughout the analysis. In Step 3 a shear force is applied to rod 1 such that $\tau_{eq} < \tau_{crit}$ and the rod remains sticking. The

shear force is removed in Step 4. In Step 5 the friction model for rod 1 is modified by providing test data. A shear force is applied such that $\tau_{eq} < \tau_{crit}$ and the rod remains sticking. The shear forces are removed in Step 6. In Step 7 the original friction model is specified with the friction properties reset to their original values. A slip is applied to rod 1. In Step 8 a new friction model is introduced for rod 2. In Step 9 a slip is applied to rod 2 in which the slip rate is varied by prescribing the displacement with an amplitude curve during the static step.

Surface interaction for rod 1:

Step 1

$\mu_s = 0.3$;

$\mu_k = 0.1$;

$d_c = 4$.

Step 5

Test data input:

$\mu_1 = 0.5, \dot{\gamma}_1 = 0.0$;

$\mu_2 = 0.3, \dot{\gamma}_2 = 0.2$;

$\mu_3 = 0.2, \dot{\gamma}_3 = \infty$.

Step 7

Same as specified in Step 1.

Step 8

$\mu = 0.0$ for the elastic slip formulation.

Rough friction model for the Lagrange multiplier formulation.

Surface interaction for rod 2:

Step 1

$\mu = 0.0$ for the elastic slip formulation.

Step 8

Test data input:

$\mu_1 = 0.3, \dot{\gamma}_1 = 0.0$;

$\mu_2 = 0.1, \dot{\gamma}_2 = 0.2$.

It is assumed that $\frac{(\mu_2 - \mu_\infty)}{(\mu_1 - \mu_\infty)} = 0.05$.

Results and discussion

Contact pressure, shear forces, and slip were verified.

Input files

Coulomb friction model:

eifricc1e.inp	Elastic slip formulation, 1 local tangent direction.
eifricc1l.inp	Lagrange multiplier formulation, 1 local tangent direction.
eifricc2e.inp	Elastic slip formulation, 2 local tangent directions.
eifricc2l.inp	Lagrange multiplier formulation, 2 local tangent directions.

Exponential decay model:

eifrice1e.inp	Elastic slip formulation, 1 local tangent direction.
eifrice1l.inp	Lagrange multiplier formulation, 1 local tangent direction.
eifrice2e.inp	Elastic slip formulation, 2 local tangent directions.
eifrice2l.inp	Lagrange multiplier formulation, 2 local tangent directions.

Friction models in Abaqus/Explicit

Products: Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

CPE3 MASS

Features tested

Friction surface interaction.

Problem description

The friction models provided in Abaqus/Explicit are tested on a simple problem, and the results are compared to analytical solutions.

The first example uses the classical Coulomb friction model. The critical shear stress, τ_{crit}, at which surfaces begin to slide with respect to each other is given by

$$\tau_{crit} = \mu p,$$

where μ is the coefficient of friction and p is the normal pressure.

The second example uses the classical Coulomb friction model explained above with softened tangential behavior. While under the condition of sticking friction, the surfaces are allowed to slip, generating "elastic" slip. The amount of elastic slip, γ_{el}, is given by

$$\gamma_{el} = \kappa\tau,$$

where κ is the slope of the shear stress versus elastic slip curve and τ is the shear stress calculated from the friction law. While under the condition of slipping friction, the behavior is identical to the classical Coulomb friction model without softened tangential behavior.

The third example uses a rate-dependent friction model in which the static friction coefficient, μ_s, decays to the kinetic friction coefficient, μ_k, according to the exponential form,

$$\mu = \mu_k + (\mu_s - \mu_k)\, e^{-d_c \dot{\gamma}_{eq}},$$

where d_c is a user-defined decay parameter and $\dot{\gamma}_{eq}$ is the slip rate. This model is referred to as the exponential decay friction model.

The fourth example uses the Coulomb friction model with dependencies to simulate slip-rate-dependent friction. The coefficient of friction is defined as a function of the slip rate and the normal contact pressure. To facilitate comparison of the analyses, the tabular data are synthesized to approximate the exponential decay model.

The fifth example uses a rough friction model with softened tangential behavior. With this model, all tangential motion is in the form of elastic slip. This model differs from the second example in that the shear stress is no longer limited by τ_{crit}, so no frictional slip can occur.

The problem consists of a rectangular block of two CPE3 elements sliding on a rigid surface. The block is 5 inches long, 1 inch high, and 1 inch thick. The elastic modulus is 3×10^7 psi, and the density is 7.3×10^{-4} lbf s^2/in^4. A uniform pressure of 2000 psi is applied on the top face of the block, and an initial velocity of 200 in/s is prescribed at each node of the block. The same problem is used to test user subroutine *VFRIC* in *VFRIC, VFRIC_COEF, and VFRICTION*.

For the classical Coulomb friction model $\mu = 0.15$; for the exponential decay friction model $\mu_s = 0.15$, $\mu_k = 0.05$, and $d_c = 0.01$ s/in; for the models including softened tangential behavior $\kappa = 10^4$ psi/in.

Results and discussion

The results for all five examples are discussed below.

Results for the classical Coulomb friction model

The prescribed external load produces a normal pressure of 2000 psi and a frictional stress of 300 psi. This corresponds to a negative acceleration of 4.110×10^5 in/s^2 in the tangential direction, since the frictional stress opposes the motion of the block. Given the initial velocity and the acceleration, the block should come to rest after sliding a distance of 4.866×10^{-2} inches over a time period of 4.866×10^{-4} s. The corresponding values for sliding distance and time period obtained with the Coulomb finite element model are 4.866×10^{-2} inches and 4.878×10^{-4} s, respectively. The numerical results show some oscillations in the normal reactions and frictional forces caused by the inertial effect of nodes on the top of the block; there is some oscillation of the block in a shear mode, even after the block stops sliding.

Results for the classical Coulomb friction model with softened tangential behavior

As in the preceding example, the critical frictional stress between the block and rigid surface is 300 psi. Elastic slip will be generated until the frictional stress exceeds the critical stress, and frictional slip will be initiated. The block then slows to zero velocity due to the frictional dissipation and reverses direction as the stored elastic slip is converted back into kinetic energy. The analytical solution for a rigid block with the given initial velocity predicts that the block will reverse its direction of travel at a time of 5.638×10^{-4} s at a distance of 6.367×10^{-2} inches. The corresponding values for time and distance from the finite element model are 5.704×10^{-4} s and 6.338×10^{-2} inches, respectively.

Results for the exponential decay friction model

In this model the velocity of a node in contact corresponds to the slip rate for the friction model. *Table 1* compares the velocity values obtained from a closed-form solution, which assumes the block to be rigid, to the average velocity of the contacting nodes in the finite element model. The differences are caused by the oscillations in the shear mode of the finite element model. The analysis using penalty contact has additional differences due to the default viscous contact damping, which contributes to the contact forces opposing the motion of the block.

Results for the Coulomb friction model with dependencies

The tabular data for this model are chosen to approximate the exponential decay model described in the previous subsection. Both slip rate and pressure dependence are included in the model to verify the code. The pressure dependence is defined such that the interpolated values at a pressure equal to 2000 psi correspond to the exponential decay model considered previously. *Table 2* compares the average velocities of the contacting nodes in the finite element model with the velocity values obtained from a closed-form solution based on a rigid block. Small differences occur as a result of oscillations in the finite element model and the linear interpolation of the tabular data.

Results for the rough friction model with softened tangential behavior

With rough frictional behavior and tangential softening (without viscous contact damping), this model essentially behaves like an undamped oscillator. The analytical solution to a point mass oscillating on a linear spring without damping gives the amplitude of the oscillation in slip to be $5.404 \times 10^{—2}$ inches and the time at which the slip direction first reverses to be 4.244×10^{-4} s. The corresponding values for amplitude and time from the finite element model using penalty contact are 5.403×10^{-2} inches and 4.273×10^{-4} s, respectively. The corresponding values for the amplitude and time from the finite element model using kinematic contact are 5.378×10^{-2} inches and 4.234×10^{-4} s, respectively.

Input files

fric_coulomb.inp	Coulomb friction model.
fric_coulomb_soft.inp	Coulomb friction model with softened tangential behavior.
fric_exponential_decay.inp	Exponential decay friction model.
fric_coulomb_dep.inp	Coulomb friction model with slip rate and pressure dependencies.
fric_coulomb_deppnlty.inp	Coulomb friction model with slip rate, pressure dependencies, and penalty contact.
fric_rough.inp	Rough friction model with softened tangential behavior and kinematic contact.
fric_rough_pnlty.inp	Rough friction model with softened tangential behavior and penalty contact.
fric_coulomb_dep.inp	Coulomb friction model with slip rate dependencies and general contact.
fric_coulomb_slipdep_gcont.inp	Coulomb friction model with slip rate dependencies and general contact.

Tables

Table 1: Comparison of velocity values for the exponential decay friction model.

Time 10^{-4} s	Velocity (analytical) in/s	Velocity (model) in/s
1.0301	181.7	181.8
2.0042	163.6	164.2
3.0001	144.1	143.5
4.0064	123.1	123.9
5.0000	100.6	100.5
6.0284	74.73	75.2
7.0022	46.87	47.98
8.0017	12.88	11.85
8.2289	4.054	2.931

Table 2: Comparison of velocity values for the Coulomb friction model with dependencies.

Time	Velocity	Velocity
10^{-4} s	(analytical) in/s	(model) in/s
1.0301	181.7	182.0
2.0042	163.6	164.4
3.0001	144.1	143.9
4.0064	123.1	124.4
5.0000	100.6	101.1
6.0284	74.73	75.98
7.0022	46.87	48.93
8.0017	12.88	13.25
8.2289	4.054	4.454

Cohesive surface interaction

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Features tested

This section provides verification for the following options:

- Cohesive behavior properties
- Material and contact properties to define damage initiation
- Material properties to define damage evolution

Cohesive behavior options

Problem description

The following usages of surface-based cohesive behavior are verified in these tests:

- Cohesive behavior properties
- Cohesive behavior restricted to initially contacting nodes
- Cohesive behavior allowing repeated contacts
- Coupled traction-separation behavior
- Cohesive behavior between beam elements
- Cohesive behavior under highly ductile deformation

Model: This test consists of six cases, each of which illustrate one of the usages of the cohesive behavior properties listed above. Each of the first four cases comprise two blocks of solid elements bonded together with cohesive surfaces defined at the interface between the blocks. In all these cases except Case 2 the initial configuration is fully compliant, with the secondary and main surfaces touching each other exactly without any overclosures or gaps. In Case 2 there is an initial gap between some nodes of the secondary surface and the main surface that is not resolved at the start of the analysis.

Case 1 has cohesive behavior defined with default parameters; hence, cohesive behavior is defined for all nodes of the secondary surface that are initially in contact with the main surface and also the secondary nodes that might come in contact later, and postfailure cohesive behavior is not defined. There are no data line values prescribed, so the default cohesive stiffness values calculated by Abaqus are used to enforce cohesive behavior. Progressive failure of the cohesive bond is modeled using the maximum stress damage initiation criterion and damage evolution with linear displacement–based softening behavior.

Case 2 has cohesive behavior defined with the initially contacting nodes. Since there is an initial gap between some nodes of the secondary surface and the main surface, these nodes are not in contact in the initial configuration and, thus, cohesive behavior is not enforced at these nodes. Uncoupled nondefault cohesive stiffness values are prescribed on the data line. No damage model is defined for this case, so the cohesive bond does not degrade and fail.

Case 3 is similar to Case 1. In addition, postfailure cohesive behavior is enforced for recurrent contacts at nodes on the secondary surface.

Case 4 has cohesive behavior with coupled traction-separation behavior. Coupled cohesive stiffness values are prescribed on the data line. Progressive failure of the cohesive bond is modeled using the maximum stress damage initiation criterion and damage evolution with linear displacement–based softening behavior.

Case 5 has cohesive behavior between two beams with circular cross-sections. The default cohesive contact stiffness is used.

Case 6 is similar to Case 4, but the cohesive constraints remain in effect after significant separation. The cohesive contact normal direction is calculated using two different methods. In the first method (default), the cohesive contact normal direction is based solely on the main surface normal direction. In the second method, the main and secondary surface normals influence the evolution of the contact normal direction equally, which is more consistent with cohesive element behavior.

Loading: The loading is the same in the first three cases: the blocks are first pulled apart in pure normal mode by applying displacement boundary conditions, then they are brought into contact, and finally they are again pulled apart. In the fourth case, a mixed mode loading is applied.

Results and discussion

The response of the cohesive surface is correct in all cases. For Case 1 once the cohesive bond breaks, no further cohesive constraints are enforced. In Case 3, which allows postfailure cohesive behavior, cohesive constraints are reinforced when the surfaces reenter contact following the first debonding.

Input files

Abaqus/Explicit input file

gcont_cohesive_options.inp	Verification test for different cohesive behavior options.

Abaqus/Standard input files

gcont_cohesive_options_std_2d.inp	Verification test for different cohesive behavior options in two dimensions.
gcont_cohesive_options_std_3d.inp	Verification test for different cohesive behavior options in three dimensions.

Damage modeling with cohesive surfaces in Abaqus/Explicit

Problem description

This test verifies damage modeling with cohesive surfaces using different damage initiation criteria and damage evolution laws to simulate the failure of cohesive layers.

The maximum separation and quadratic stress damage initiation criteria are used. Damage evolution is defined based on either effective displacement or energy dissipated. Linear, exponential, and tabular softening laws are defined to specify the nature of the evolution of the damage variable. Each damage model is verified for damage in pure normal and two pure shear modes (one shear mode for two-dimensional and axisymmetric elements). The dependence of damage evolution on the mode mix measure specified in tabular, power law, or Benzeggagh-Kenane form is also considered in this test.

Results and discussion

Degradation of the response of the cohesive surfaces begins when the specified damage initiation criterion is met. The damage variable evolves according to the evolution law specified in terms of displacement or energy dissipation.

Input files

Abaqus/Explicit input files

gcont_mxe_damdisp_softlin_xpl.inp	MAXU damage initiation, displacement-based damage evolution with LINEAR softening for cohesive surfaces.
gcont_qds_damdisp_softlin_xpl.inp	QUADS damage initiation, displacement-based damage evolution with LINEAR softening for cohesive surfaces.
gcont_mxe_damdisp_softexp_xpl.inp	MAXU damage initiation, displacement-based damage evolution with EXPONENTIAL softening for cohesive surfaces.
gcont_qds_damdisp_softexp_xpl.inp	QUADS damage initiation, displacement-based damage evolution with EXPONENTIAL softening for cohesive surfaces.
gcont_mxe_damdisp_softtab_xpl.inp	MAXU damage initiation, displacement-based damage evolution with TABULAR softening for cohesive surfaces.
gcont_qds_damdisp_softtab_xpl.inp	QUADS damage initiation, displacement-based damage evolution with TABULAR softening for cohesive surfaces.
gcont_mxe_damener_softlin_xpl.inp	MAXU damage initiation, energy-based damage evolution with LINEAR softening for cohesive surfaces.
gcont_qds_damener_softlin_xpl.inp	QUADS damage initiation, energy-based damage evolution with LINEAR softening for cohesive surfaces.
gcont_mxe_damener_softexp_xpl.inp	MAXU damage initiation, energy-based damage evolution with EXPONENTIAL softening for cohesive surfaces.
gcont_qds_damener_softexp_xpl.inp	QUADS damage initiation, energy-based damage evolution with EXPONENTIAL softening for cohesive surfaces.
gcont_damdisp_mixtrac_xpl.inp	Displacement-based damage evolution with traction-dependent mode mix measure for cohesive surfaces.
gcont_damdisp_mixener_xpl.inp	Displacement-based damage evolution with energy-dependent mode mix measure for cohesive surfaces.
gcont_damener_mixtrac_xpl.inp	Energy-based damage evolution with traction-dependent mode mix measure for cohesive surfaces.
gcont_damener_mixener_xpl.inp	Energy-based damage evolution with energy-dependent mode mix measure for cohesive surfaces.
gcont_damener_mixener_xpl.inp	Energy-based damage evolution with energy-dependent mode mix measure and rate-dependent damage initiation with logarithmic interpolation for cohesive surfaces.
gcont_damener_mixener_xpl.inp	Energy-based damage evolution with energy-dependent mode mix measure and rate-dependent damage initiation with linear interpolation for cohesive surfaces.
gcont_damener_mixener_xpl.inp	Energy-based damage evolution with energy-dependent mode mix measure and rate-dependent damage initiation with logarithmic interpolation for cohesive surfaces.
gcont_damener_mixener_xpl.inp	Energy-based damage evolution with energy-dependent mode mix measure and rate-dependent damage initiation with linear interpolation for cohesive surfaces.

gcont_damdisp_mixener_xpl_rateDep_loginterp.inp	Energy-based damage evolution with energy-dependent mode mix measure and rate-dependent damage initiation with logarithmic interpolation for cohesive surfaces.
gcont_damdisp_mixener_xpl_rateDep_loginterp2.inp	Energy-based damage evolution with energy-dependent mode mix measure and rate-dependent damage initiation with logarithmic interpolation for cohesive surfaces.
gcont_damdisp_mixener_xpl_rateDep_lininterp2.inp	Energy-based damage evolution with energy-dependent mode mix measure and rate-dependent damage initiation with linear interpolation for cohesive surfaces.

Abaqus/Standard input files

gcont_mxe_damdisp_softlin_std.inp	MAXU damage initiation, displacement-based damage evolution with LINEAR softening for cohesive surfaces.
gcont_qds_damdisp_softlin_std.inp	QUADS damage initiation, displacement-based damage evolution with LINEAR softening for cohesive surfaces.
gcont_mxe_damdisp_softexp_std.inp	MAXU damage initiation, displacement-based damage evolution with EXPONENTIAL softening for cohesive surfaces.
gcont_qds_damdisp_softexp_std.inp	QUADS damage initiation, displacement-based damage evolution with EXPONENTIAL softening for cohesive surfaces.
gcont_mxe_damdisp_softtab_std.inp	MAXU damage initiation, displacement-based damage evolution with TABULAR softening for cohesive surfaces.
gcont_qds_damdisp_softtab_std.inp	QUADS damage initiation, displacement-based damage evolution with TABULAR softening for cohesive surfaces.
gcont_mxe_damener_softlin_std.inp	MAXU damage initiation, energy-based damage evolution with LINEAR softening for cohesive surfaces.
gcont_qds_damener_softlin_std.inp	QUADS damage initiation, energy-based damage evolution with LINEAR softening for cohesive surfaces.
gcont_mxe_damener_softexp_std.inp	MAXU damage initiation, energy-based damage evolution with EXPONENTIAL softening for cohesive surfaces.
gcont_qds_damener_softexp_std.inp	QUADS damage initiation, energy-based damage evolution with EXPONENTIAL softening for cohesive surfaces.
gcont_damdisp_mixtrac_std.inp	Displacement-based damage evolution with traction-dependent mode mix measure for cohesive surfaces.
gcont_damdisp_mixener_std.inp	Displacement-based damage evolution with energy-dependent mode mix measure for cohesive surfaces.
gcont_damener_mixtrac_std.inp	Energy-based damage evolution with traction-dependent mode mix measure for cohesive surfaces.
gcont_damener_mixener_std.inp	Energy-based damage evolution with energy-dependent mode mix measure for cohesive surfaces.

Breakable ties with cohesive surfaces

Problem description

This test verifies modeling “breakable ties” using cohesive behavior and progressive damage. A box and its lid, both modeled with solid elements, are tied together via cohesive behavior at the interface. Default cohesive behavior options are used. The bottom of the box is fixed using prescribed boundary conditions, while the lid is pulled apart via prescribed displacements applied through a kinematic coupling acting on the top surface of the lid.

The maximum stress damage initiation criterion is used. Damage evolution is defined using effective displacement with a linear softening law.

Results and discussion

This test verifies modeling of “breakable ties” using cohesive surfaces. Degradation of the response of the cohesive surfaces begins when the specified damage initiation criterion is met. The damage variable evolves according to the evolution law specified.

Input files

Abaqus/Explicit input file

gcont_cohbehv_tiebreak.inp	Verification test for tie break using cohesive surfaces.

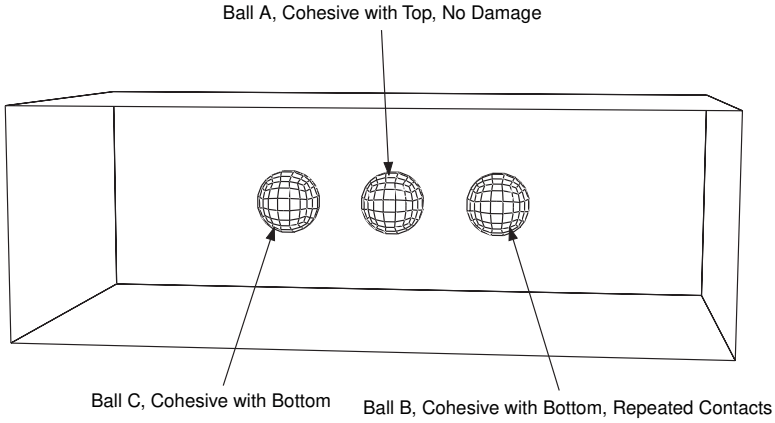
Abaqus/Standard input file

gcont_cohbehv_tiebreak_std.inp	Verification test for tie break using cohesive surfaces.

Sticky contact with cohesive surfaces

Problem description

This test verifies modeling “sticky contact” using cohesive behavior and progressive damage. A box, modeled as a rigid body, contains three balls that are modeled using shell elements. The box is completely fixed; the balls, initially suspended in the gap between the top and bottom walls of the rigid box, are given identical initial velocities resulting in their simultaneous impact with the bottom wall of the box. The behavior of each of the balls (Ball A, Ball B, and Ball C) is described below.



Ball A has cohesive behavior without progressive damage defined between its surface and the top wall of the box. No cohesive stiffness is specified, and the default values are used. When this ball impacts the bottom wall, it does not experience any cohesive forces, since no cohesive behavior is prescribed for the interaction between this ball and the bottom wall. The ball rebounds and strikes the top wall of the box, where cohesive forces act to prevent it from rebounding again and ensure that it remains stuck to the top wall for the rest of the analysis.

Ball B has cohesive behavior with progressive damage defined between its surface and the bottom wall of the box. No cohesive stiffness is specified, and the default values are used. The damage model uses the maximum stress damage initiation criteria and has damage evolution defined based on effective displacement with a linear

softening law. In addition, postfailure cohesive behavior is enforced for recurrent contacts at nodes on the secondary surface. When this ball impacts the bottom wall and tries to rebound, the cohesive forces act to restrain it from rebounding. However, since the elastic energy of the collision is high, eventually damage initiates, ultimate failure occurs, and the ball breaks free. It then goes on to hit the top wall. There is no cohesive behavior defined with the top wall, so Ball B does not experience any cohesive forces and bounces back and impacts the bottom wall again. Since postfailure cohesive behavior is allowed, cohesive forces reactivate when the ball attempts to rebound again. However, on second impact, the momentum and kinetic energy of the ball is considerably less than during first impact, owing to the dissipation that occurred due to the damage work done during first impact. The cohesive forces this time are sufficiently high to restrain it from rebounding again, and the ball remains stuck to the bottom wall for the rest of the analysis.

Ball C has exactly the same cohesive behavior and progressive damage defined between its surface and the bottom wall as Ball B. As with Ball B, when this ball impacts the bottom wall and tries to rebound, the cohesive forces act to restrain it from rebounding. However, since the elastic energy of the collision is high, eventually damage initiates, ultimate failure occurs, and the ball breaks free. It then goes on to hit the top wall. There is no cohesive behavior defined with the top wall, so Ball C does not experience any cohesive forces and bounces back and impacts the bottom wall again. Since no postfailure cohesive behavior is allowed, cohesive forces are not activated when the ball attempts to rebound following the second impact with the bottom wall. The ball rebounds again and keeps bouncing back and forth between the top and bottom walls throughout the rest of the analysis.

Results and discussion

This test verifies modeling of “sticky contact” using cohesive surfaces. Degradation of the response of the cohesive surfaces begins when the specified damage initiation criterion is met. The damage variable evolves according to the evolution law specified.

Input files

Abaqus/Explicit input file

gcont_cohbehv_stickycont.inp	Verification test for sticky contact.

Abaqus/Standard input file

gcont_cohbehv_stickycont_std.inp	Verification test for sticky contact.

Input files required for both Abaqus/Explicit and Abaqus/Standard

tennis_ef1.inp	First file for the sticky contact verification test.
tennis_ef2.inp	Second file for the sticky contact verification test.

Cohesive contact between beams in Abaqus/Explicit

Problem description

This test verifies cohesive contact behavior between two beams with circular cross-sections. *#unique_4512/unique_4512_Connect_42_BeamCohesionInitConfig* shows the initial configuration for two beams with circular cross-sections cohered where they touch. Each beam is modeled with five beam elements (of type B31) and linear elastic material behavior. Default cohesive contact stiffness is in effect. Both ends of the gray beam are fixed, while both ends of the blue beam are moved such as to develop tensile and shear stress in the cohesive bond.

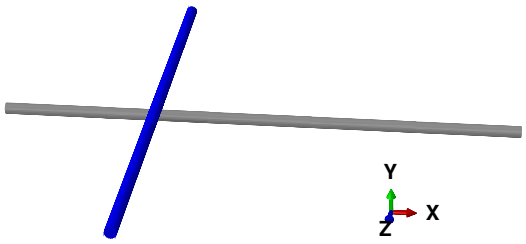


Figure 1: Initial configuration of two cohered circular beams that are just touching at a point.

Results and discussion

The left and right sides of *Figure 2* show cohesive contact normal and shear force vectors, respectively, in the deformed configuration. These force vectors act on nodes of edges involved in the edge-to-edge cohesive bond to resist relative motion at the bond location.

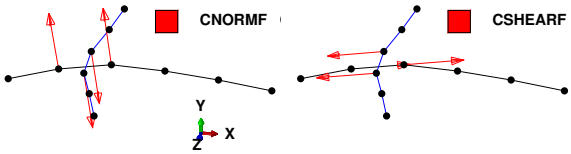


Figure 2: Vector plots of normal and shear cohesive contact forces in the final configuration of two cohered beams.

Input files

Abaqus/Explicit input file

gcont_cohesive_2beams_pullshear_xpl.inp	Two circular beams with cohesive contact.

Modeling of cohesive surfaces in Abaqus/Explicit under highly ductile deformation

Problem description

This test verifies cohesive contact behavior involving ductile cohesive bonds. *Figure 3* shows the initial and deformed configurations of a peeling test.

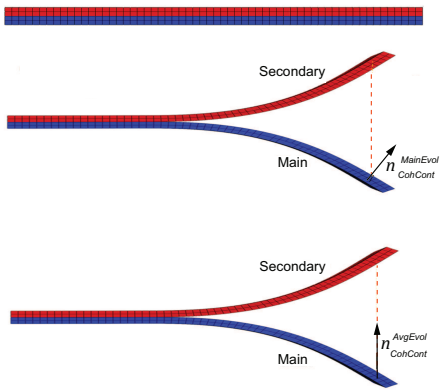


Figure 3: Peeling test model under large, highly ductile deformation.

The initial configuration of the model (top image) has two flat regions that are initially touching and bonded with ductile cohesive contact along their entire length. Each region has vertical symmetry in the geometry and mesh refinement. The left end of the model is held fixed, while the right ends are pulled apart symmetrically in equal and opposite directions up to a maximum relative displacement of 20 mm. Cohesive damage is defined such that the initiation occurs at a small opening spacing but evolves to complete failure at an effective cohesive strain magnitude of 20 mm.

In one version of the model (shown in the middle image), the cohesive contact normal direction evolves with the main surface, which is the default behavior. In this case, the cohesive strain has normal and shear components.

In another version of the model (shown in the bottom image), the cohesive contact normal direction evolves with the average evolution of the main and secondary surface normal directions. In this case, the cohesive strain is aligned with the cohesive contact normal direction.

Results and discussion

Figure 4 compares plots of COPEN for the default and nondefault options for evolution of the cohesive contact normal when the maximum relative vertical displacement reaches a value of 20 mm.

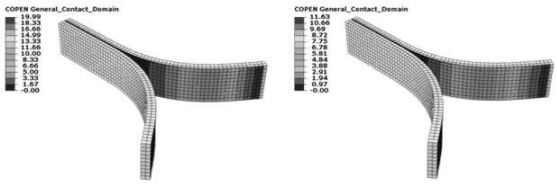


Figure 4: Tensile cohesive normal strain (COPEN) for the peeling test with a maximum cohesive strain magnitude of 20 mm using the nondefault option (left) and the default option (right) for the cohesive contact normal direction evolution.

Using the nondefault evolution of the cohesive contact normal direction, COPEN reaches a value very close to 20 mm on the nodes at the right end. Using the default evolution of the cohesive contact normal direction, the maximum normal cohesive strain component is 11.63 mm (implying that the cohesive shear strain magnitude is 16.27 mm). The cohesive normal strain is 58% smaller using the default method compared to the nondefault method.

Figure 5 shows a plot of the tensile contact normal stresses (CPRESS) for the model.

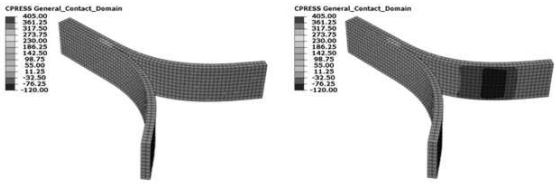


Figure 5: Tensile cohesive normal pressures (CPRESS) for the peeling test with a maximum cohesive strain magnitude of 20 mm using the nondefault option (left) and the default option (right) for the cohesive contact normal direction evolution.

There are significant differences in the tensile contact pressure profiles between the default and nondefault methods. Using the nondefault method, the tensile contact normal pressures are much smaller with larger opening spacing, which is consistent with the linear damage evolution behavior defined under monotonic loading. The tensile contact pressures for the default method show the maximum values at opening spacings around 10 mm. The maximum tensile contact pressure value is about 600% larger for the default method compared to the nondefault method.

Under high ductile cohesive deformation, much higher accuracy is achieved if the contact normal direction is calculated based on the evolution of both the main and secondary surface normal directions.

Input files

Abaqus/Explicit input file

gcont_cohesive_peeling_xpl.inp	Peeling test with cohesive contact under highly ductile deformation using the default method of evolving the contact normal direction with the main surface.
gcont_cohesive_peeling_ avgnormevol_xpl.inp	Peeling test with cohesive contact under highly ductile deformation in which the contact normal direction evolves with both the main and secondary normal directions.

Modified contact pressure-overclosure relationship

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

CAX4 CAX4H CAX4R IRS21A ISL21A GAPUNI

Features tested

Modifications to the contact pressure-overclosure relationship are tested for a number of cases.

Problem description

This set of tests verifies the softened contact option. All the tests are for axisymmetric, large-displacement, static analyses with finite sliding. The model in each test consists of a die pressing down on a rubber cylinder. The die is modeled either as a rigid surface or as a deformable body with mild steel properties. CAX4 elements are used in the Abaqus/Standard analyses, and CAX4R elements are used in the Abaqus/Explicit analyses. The blank is modeled as an incompressible Mooney-Rivlin material with CAX4H elements in Abaqus/Standard and CAX4R elements in Abaqus/Explicit. The bottom surface of the blank is constrained against vertical motion.

Each analysis has one step, in which a vertical prescribed displacement is applied to the die.

Results and discussion

The analysis results indicate that the die penetrates the blank according to the prescribed pressure-overclosure relationship.

Input files

Abaqus/Standard input files

Tabular pressure-overclosure relationship:

xsctdd.inp	Deformable-deformable surface-based contact.
xsctdr.inp	Deformable-rigid surface-based contact.
xsctirs.inp	IRS contact elements.
xsctisl.inp	ISL contact elements.
xsctgap.inp	GAPUNI contact elements.

Linear pressure-overclosure relationship:

xscldr.inp	Deformable-rigid surface-based contact.

Exponential pressure-overclosure relationship:

xscxdd.inp	Deformable-deformable surface-based contact.
xscxdr.inp	Deformable-rigid surface-based contact.

xscxirs.inp	IRS contact elements.
xscxisl.inp	ISL contact elements.
xscxgap.inp	GAPUNI contact elements.

Abaqus/Explicit input files

Linear pressure-overclosure relationship:

linsoft_xpl_nosep.inp	Deformable-rigid kinematic contact with no separation surface behavior.
linsoft_xpl_nosep_pnlty.inp	Deformable-rigid penalty contact with no separation surface behavior.

Exponential pressure-overclosure relationship:

expsoft_xpl_def_pnlty.inp	Deformable-deformable penalty contact.
expsoft_xpl_def_gc.inp	Deformable-deformable general contact.
expsoft_xpl_rig_pnlty.inp	Deformable-rigid penalty contact.
expsoft_xpl_rig_gc.inp	Deformable-rigid general contact.

Tabular pressure-overclosure relationship:

tabsoft_xpl_def.inp	Deformable-deformable kinematic contact.
tabsoft_xpl_def_gc.inp	Deformable-deformable general contact.
tabsoft_xpl_rig.inp	Deformable-rigid kinematic contact.
tabsoft_xpl_rig_gc.inp	Deformable-rigid general contact.

Interference fit and specified clearance

In this section:

- *Contact with time-dependent prescribed interference values*
- *Contact with initial overclosure of curved surfaces*
- *Small-sliding contact with specified clearance or overclosure values*

Contact with time-dependent prescribed interference values

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

CPE4 C3D8

Features tested

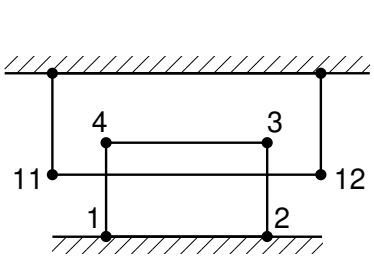
Contact interference

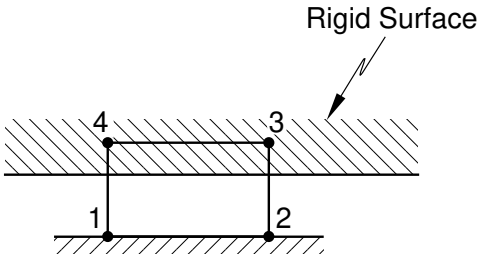
Surface on a deformable body, surface on a deformable body or a rigid surface, and magnitude of allowable interference

Problem description

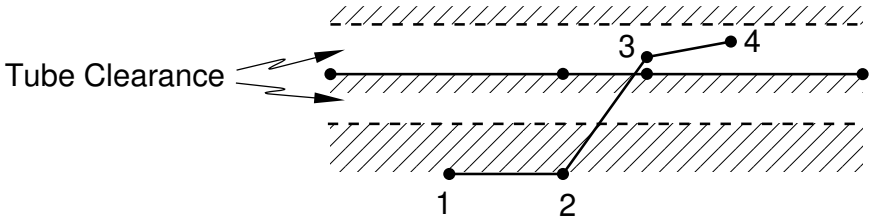
The tests exercise the three ways in which contact interference can be used. Either a simple amount of allowable interference is specified, an allowable interference along a prescribed direction is specified, or the automatic shrink fit procedure is invoked. In this latter case Abaqus initializes the amount of allowable interference at each contact point with the penetration it calculates at the beginning of the analysis.

Most of the models consist of two elements lying next to each other with their contact surfaces initially interfering by a magnitude of 0.2 units. In the case of rigid surfaces there is only one element initially interfering with a straight rigid surface. The solid elements are either 4-node quads or 8-node bricks, as a substrate for the appropriate contact elements. A contact interference of magnitude 0.2 units is used to resolve the interference in (typically) five increments.





In the case of tube within tube elements (ITT) the model consists of two beams at a variable transverse distance from each other. One is totally fixed, and the other is fixed only axially. An initial tube clearance of 0.5 units produces interferences of up to 0.5 units. This interference is resolved by using the contact interference definition with a magnitude of 0.5 units.



Material:

Solid

Young's modulus	1.0×10^5
Poisson's ratio	0.0
Conductivity	5.0
Density	0.5
Specific heat	0.3

Interface

Friction coefficient	0.0
Gap conductance	2.0 (coupled temperature-displacement elements)

Results and discussion

The interference is resolved in five increments.

Input files

Surface-based contact

Allowable interference:

ei34siis.inp	C3D8 elements, small-sliding.
eig1siis.inp	C3D8 elements, small-sliding, node-based surface.
ei34siisf.inp	C3D8 elements, finite-sliding.
ei31siisf.inp	C3D8 elements, finite-sliding, node-based surface.
ei22siis.inp	CPE4 elements, small-sliding.
ei22ssis.inp	CPE4 elements, finite-sliding.
eip1sris.inp	CPE4 elements, analytical rigid surface.

Allowable interference along a prescribed direction:

ei34siid.inp	C3D8 elements, small-sliding.
ei34srid.inp	C3D8, R3D4 elements.
eig1siid.inp	C3D8 elements, small-sliding, node-based surface.
ei34siidf.inp	C3D8 elements, finite-sliding.
ei31siidf.inp	C3D8 elements, finite-sliding, node-based surface.

ei22siid.inp CPE4 elements, small-sliding.
ei22ssid.inp CPE4 elements, finite-sliding.
eip1srid.inp CPE4 elements, analytical rigid surface.

Automatic shrink fit:

ei22siif.inp CPE4 elements, small-sliding.
ei22ssif.inp CPE4 elements, finite-sliding.
ei34siiff.inp C3D8 elements, finite-sliding.

Contact element approach (undocumented)

Allowable interference:

ei21stvs.inp B21, ITT21 elements.
eis1sgvs.inp C3D8, GAPSPHER elements.
eiu1sgvs.inp CPE4, GAPUNI elements.

Allowable interference along a prescribed direction:

ei21stvd.inp B21, ITT21 elements.
eis1sgvd.inp C3D8, GAPSPHER elements.
eiu1sgvd.inp CPE4, GAPUNI elements.

Contact with initial overclosure of curved surfaces

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D8 C3D8R CPS4 CPS4R

Features tested

Contact damping

Contact pair

Modified contact pressure-overclosure relationship

Problem description

The model consists of two concentric rings with a small initial overclosure. In Abaqus/Standard the initial overclosure is resolved during a static step. In Abaqus/Explicit the initial overclosure is resolved during the first step of a dynamic simulation.

The two-dimensional model consists of two 16-element rings, and the three-dimensional model consists of two 32-element rings. The elements of the inner and outer rings are perfectly aligned.

Material:

Young's modulus	206800
Poisson's ratio	0.32
Density	8.01×10^{-6}
Interface friction coefficient	0.0

Results and discussion

The interference is resolved for models using hard contact. In the case of softened contact the interference is reduced until equilibrium is reached; any residual overclosure at the end of the step can be reduced by increasing the stiffness of the pressure-overclosure relationship.

Input files

Abaqus/Standard input files

ei24siso.inp	CPS4 elements, small-sliding.
ei24siso_surf.inp	CPS4 elements, small-sliding, surface-to-surface constraint enforcement method.
ei24ssso.inp	CPS4 elements, finite-sliding, **CONTACT INTERFERENCE* with SHRINK.

ei38siso.inp	C3D8 elements, small-sliding.
ei38siso_surf.inp	C3D8 elements, small-sliding, surface-to-surface constraint enforcement method.
ei34sfso.inp	C3D8 elements, finite-sliding.
ei34sfso_1.inp	C3D8 elements, finite-sliding with HCRIT and SMOOTH parameters.
ei34sfso_2.inp	C3D8 elements, finite-sliding, **SURFACE BEHAVIOR* with NO SEPARATION.
ei34sfso_3.inp	C3D8 elements, finite-sliding, **SURFACE BEHAVIOR* with PRESSURE-OVERCLOSURE=EXPONENTIAL.
ei34sfso_4.inp	C3D8 elements, finite-sliding, **SURFACE BEHAVIOR* with PRESSURE-OVERCLOSURE=TABULAR.
ei34sfso_5.inp	C3D8 elements, finite-sliding, **CONTACT DAMPING* with DEFINITION=DAMPING COEFFICIENT.

Abaqus/Explicit input files

interference2d_xpl_finite.inp	CPS4R elements, finite-sliding kinematic contact, **SURFACE BEHAVIOR* with PRESSURE-OVERCLOSURE=TABULAR.
interference2d_xpl_finite_pnlty.inp	CPS4R elements, finite-sliding penalty contact, **SURFACE BEHAVIOR* with PRESSURE-OVERCLOSURE=TABULAR.
interference3d_xpl_finite.inp	C3D8R elements, finite-sliding kinematic contact, **SURFACE BEHAVIOR* with PRESSURE-OVERCLOSURE=TABULAR.
interference3d_xpl_finite_pnlty.inp	C3D8R elements, finite-sliding penalty contact, **SURFACE BEHAVIOR* with PRESSURE-OVERCLOSURE=TABULAR.
interference3d_xpl_finite_c3d8.inp	C3D8 elements, finite-sliding kinematic contact, **SURFACE BEHAVIOR* with PRESSURE-OVERCLOSURE=TABULAR.
interference3d_xpl_finite_pnlty_c3d8.inp	C3D8 elements, finite-sliding penalty contact, **SURFACE BEHAVIOR* with PRESSURE-OVERCLOSURE=TABULAR.
interference3d_xpl_nosep_cdf.inp	C3D8R elements, finite-sliding kinematic contact, **SURFACE BEHAVIOR* with PRESSURE-OVERCLOSURE=TABULAR and NO SEPARATION, **CONTACT DAMPING* with DEFINITION=CRITICAL DAMPING FRACTION.
interference3d_xpl_nosep_dc.inp	C3D8R elements, finite-sliding kinematic contact, **SURFACE BEHAVIOR* with PRESSURE-OVERCLOSURE=TABULAR and NO SEPARATION, **CONTACT DAMPING* with DEFINITION=DAMPING COEFFICIENT.
interference3d_xpl_small.inp	C3D8R elements, small-sliding kinematic contact, **SURFACE BEHAVIOR* with PRESSURE-OVERCLOSURE=TABULAR.

Small-sliding contact with specified clearance or overclosure values

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D8 C3D8R CAX4R CPE4 CPE4R R2D2 R3D4 S4R

Features tested

Contact interference

Initial clearance

Small-sliding contact pair

Problem description

The Abaqus/Standard model consists of two bodies with their contact surfaces initially overclosed. This initial overclosure is maintained throughout the analysis by specifying the initial clearance value as zero, except when contact interference is used. In these cases the initial overclosure is resolved before the load is applied.

Model:

2D solid element dimensions	$1.0 \times 1.0 \times 1.0$
3D solid element dimensions	1.0 unit in each direction
3D shell element dimensions	20 units in diameter

Material:

Young's modulus	30×10^6
Poisson's ratio	0.0
Friction coefficient	0.0

General: The Abaqus/Explicit model consists of three deformable bodies that are in contact with a rigid surface. Two different methods are used to define initial clearance values: specifying the clearance value or specifying secondary nodes and their corresponding initial clearance values on data lines that either follow the keyword line or are read from an input file. A two-dimensional model is considered with the deformable bodies modeled using CPE4R elements and the rigid body modeled using rigid elements, R2D2.

Results and discussion

In all cases the results are correct throughout the analysis.

Input files

Abaqus/Standard models involving *CLEARANCE, BOLT

contact_s4r_clear_bolt.inp	S4R elements, contact directions redefined by specifying the thread geometry data and the two points on the axis of the bolt/bolt hole.
clearance_bolt_compression.inp	C3D8R elements, contact directions redefined by specifying the thread geometry data and the two points on the axis of the bolt/nut hole; using uniform axial component normal adjustment for a right-handed bolt; the bolt under compression loading; the bolt surface acting as the main surface and the nut surface acting as the secondary surface.
clearance_bolt_tension.inp	C3D8R elements, contact directions redefined by specifying the thread geometry data and the two points on the axis of the bolt/nut hole; using uniform axial component normal adjustment for a right-handed bolt; the bolt under tension loading; the bolt surface acting as the main surface and the nut surface acting as the secondary surface.
clearance_bolt_tension_opp_ms.inp	C3D8R elements, contact directions redefined by specifying the thread geometry data and the two points on the axis of the bolt/nut hole; using uniform axial component normal adjustment for a right-handed bolt; the bolt under tension loading; the bolt surface acting as the secondary surface and the nut surface acting as the main surface.
clearance_bolt_tension_twoway.inp	C3D8R elements, contact directions redefined by specifying the thread geometry data and the two points on the axis of the bolt/nut hole; using location-dependent normal adjustment for a right-handed bolt; the bolt under tension loading; the bolt surface acting as the main surface and the nut surface acting as the secondary surface.
clearance_bolt_tension_twoway_left.inp	C3D8R elements, contact directions redefined by specifying the thread geometry data and the two points on the axis of the bolt/nut hole; using location-dependent normal adjustment for a left-handed bolt; the bolt under tension loading; the bolt surface acting as the main surface and the nut surface acting as the secondary surface.
clearance_bolt_tension_twoway_opp_ms.inp	C3D8R elements, contact directions redefined by specifying the thread geometry data and the two points on the axis of the bolt/nut hole; using location-dependent normal adjustment for a right-handed bolt; the bolt under tension loading; the bolt surface acting as the secondary surface and the nut surface acting as the main surface.
clearance_bolt_tension_wrongdir.inp	C3D8R elements, contact directions redefined by specifying the thread geometry data and the two points on the axis of the bolt/nut hole; contact lost by using the axis defined from the bolt head toward the bolt tip; using uniform axial component normal adjustment for a right-handed bolt; the bolt surface acting as the main surface and the nut surface acting as the secondary surface.
clearance_bolt_thermalexpansion_twoway.inp	C3D8R elements, contact directions redefined by specifying the thread geometry data and the two points on the axis of the bolt/nut hole; using location-dependent normal adjustment for a right-handed bolt; the bolt under thermal expansion loading; the bolt surface acting as the main surface and the nut surface acting as the secondary surface.
clearance_bolt_tension_coarse.inp	C3D8R elements, contact directions redefined by specifying the thread geometry data and the two points on the axis of the bolt/nut hole; using uniform axial component normal adjustment for a right-handed bolt; the bolt under tension loading; the bolt surface acting as the main surface and the nut surface acting as the secondary surface; coarse meshes.

clearance_bolt_tension_twoway_coarse.inp	C3D8R elements, contact directions redefined by specifying the thread geometry data and the two points on the axis of the bolt/nut hole; using location-dependent normal adjustment for a right-handed bolt; the bolt under tension loading; the bolt surface acting as the main surface and the nut surface acting as the secondary surface; coarse meshes.
clearance_bolt_compression_twoway_2d.inp	CAX4R elements, contact directions redefined by specifying the thread geometry data and the two points on the axis of the bolt/nut hole; using location-dependent normal adjustment; the bolt under compression loading; the bolt surface acting as the main surface and the nut surface acting as the secondary surface.
clearance_bolt_tension_2d.inp	CAX4R elements, contact directions redefined by specifying the thread geometry data and the two points on the axis of the bolt/nut hole; using uniform axial component normal adjustment; the bolt under tension loading; the bolt surface acting as the main surface and the nut surface acting as the secondary surface.
clearance_bolt_tension_twoway_2d.inp	CAX4R elements, contact directions redefined by specifying the thread geometry data and the two points on the axis of the bolt/nut hole; using location-dependent normal adjustment; the bolt under tension loading; the bolt surface acting as the main surface and the nut surface acting as the secondary surface.
clearance_bolt_tension_wrongdir_2d.inp	CAX4R elements, contact directions redefined by specifying the thread geometry data and the two points on the axis of the bolt/nut hole; contact lost by using the axis defined from the bolt head toward the bolt tip; using uniform axial component normal adjustment; the bolt under tension loading; the bolt surface acting as the main surface and the nut surface acting as the secondary surface.
clearance_bolt_thermalexpansion_twoway_2d.inp	CAX4R elements, contact directions redefined by specifying the thread geometry data and the two points on the axis of the bolt/nut hole; using location-dependent normal adjustment; the bolt under thermal expansion loading; the bolt surface acting as the main surface and the nut surface acting as the secondary surface.

Other two-dimensional Abaqus/Standard models

ei22siam.inp	CPE4 elements, analytical rigid surface, contact directions redefined.
ei22siam_surf.inp	CPE4 elements, analytical rigid surface, contact directions redefined, surface-to-surface constraint enforcement method.
ei22siao.inp	CPE4 elements, analytical rigid surface, contact directions calculated by Abaqus.
ei22siao_surf.inp	CPE4 elements, analytical rigid surface, contact directions calculated by Abaqus, surface-to-surface constraint enforcement method.
ei22sinm.inp	CPE4 elements, node-based surface, contact directions redefined.
ei22sinm_surf.inp	CPE4 elements, node-based surface, contact directions redefined, surface-to-surface constraint enforcement method.
ei22sino.inp	CPE4 elements, node-based surface, contact directions calculated by Abaqus.
ei22sino_surf.inp	CPE4 elements, node-based surface, contact directions calculated by Abaqus, surface-to-surface constraint enforcement method.
ei22sirc.inp	CPE4, R2D2 elements, **CONTACT INTERFERENCE*.
ei22sirc_surf.inp	CPE4, R2D2 elements, **CONTACT INTERFERENCE*, surface-to-surface constraint enforcement method.
ei22sirm.inp	CPE4, R2D2 elements, contact directions redefined.
ei22sirm_surf.inp	CPE4, R2D2 elements, contact directions redefined, surface-to-surface constraint enforcement method.
ei22siro.inp	CPE4, R2D2 elements, contact directions calculated by Abaqus.
ei22siro_surf.inp	CPE4, R2D2 elements, contact directions calculated by Abaqus, surface-to-surface constraint enforcement method.

ei22sism.inp	CPE4 elements, contact directions redefined.
ei22sism_surf.inp	CPE4 elements, contact directions redefined, surface-to-surface constraint enforcement method.
ei22siso.inp	CPE4 elements, contact directions calculated by Abaqus.
ei22siso_surf.inp	CPE4 elements, contact directions calculated by Abaqus, surface-to-surface constraint enforcement method.

Other three-dimensional Abaqus/Standard models

ei34siam.inp	C3D8 elements, analytical rigid surface, contact directions redefined.
ei34siam_surf.inp	C3D8 elements, analytical rigid surface, contact directions redefined, surface-to-surface constraint enforcement method.
ei34siao.inp	C3D8 elements, analytical rigid surface, contact directions calculated by Abaqus.
ei34siao_surf.inp	C3D8 elements, analytical rigid surface, contact directions calculated by Abaqus, surface-to-surface constraint enforcement method.
ei34sinm.inp	C3D8 elements, node-based surface, contact directions redefined.
ei34sinm_surf.inp	C3D8 elements, node-based surface, contact directions redefined, surface-to-surface constraint enforcement method.
ei34sino.inp	C3D8 elements, node-based surface, contact directions calculated by Abaqus.
ei34sino_surf.inp	C3D8 elements, node-based surface, contact directions calculated by Abaqus, surface-to-surface constraint enforcement method.
ei34sirc.inp	C3D8, R3D4 elements, **CONTACT INTERFERENCE*.
ei34sirc_surf.inp	C3D8, R3D4 elements, **CONTACT INTERFERENCE*, surface-to-surface constraint enforcement method.
ei34sirm.inp	C3D8, R3D4 elements, contact directions redefined.
ei34sirm_surf.inp	C3D8, R3D4 elements, contact directions redefined, surface-to-surface constraint enforcement method.
ei34siro.inp	C3D8, R3D4 elements, contact directions calculated by Abaqus.
ei34siro_surf.inp	C3D8, R3D4 elements, contact directions calculated by Abaqus, surface-to-surface constraint enforcement method.
ei34siro_po.inp	**POST OUTPUT* analysis.
ei34siro_po_surf.inp	**POST OUTPUT* analysis, surface-to-surface constraint enforcement method.
ei34sisc.inp	C3D8 elements, **CONTACT INTERFERENCE*.
ei34sisc_surf.inp	C3D8 elements, **CONTACT INTERFERENCE*, surface-to-surface constraint enforcement method.
ei34sism.inp	C3D8 elements, contact directions redefined.
ei34sism_surf.inp	C3D8 elements, contact directions redefined, surface-to-surface constraint enforcement method.
ei34siso.inp	C3D8 elements, contact directions calculated by Abaqus.
ei34siso_surf.inp	C3D8 elements, contact directions calculated by Abaqus, surface-to-surface constraint enforcement method.
contact_s4r_clear.inp	S4R elements, contact directions redefined by specifying the components of the vectors directly.

Abaqus/Explicit model

contact2D_clear.inp	Two-dimensional contact analysis with three contact pairs with specified initial clearances.
contact2D_clear_data.inp	File containing a list of secondary nodes and their corresponding initial clearance values.

Contact formulations

In this section:

- *Small-sliding contact between stress/displacement elements*
- *Small-sliding contact between coupled temperature-displacement surfaces*
- *Small-sliding contact between coupled thermal-electrical-structural surfaces*
- *Small-sliding contact between coupled pore pressure-displacement elements*
- *Finite-sliding contact between stress/displacement elements*
- *Finite-sliding contact between a deformable body and a rigid surface*
- *Finite-sliding contact between a deformable body and a meshed rigid surface*
- *Finite-sliding contact between coupled temperature-displacement elements*
- *Finite-sliding contact between coupled thermal-electrical-structural elements*
- *Finite-sliding contact between coupled pore pressure-displacement elements*
- *Finite sliding between concentric cylinders—axisymmetric and CAXA models*
- *Self-contact of finite-sliding deformable surfaces*
- *Contact searching for analytical rigid surfaces*
- *Surface-to-surface approach for finite-sliding contact*
- *General contact in Abaqus/Standard*

Small-sliding contact between stress/displacement elements

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D4 C3D5 C3D8 C3D8H C3D10 C3D10HS C3D10M C3D10MH
C3D20 C3D20H C3D27
CAX4 CAX4H CAX6M CAX6MH
CGAX3 CGAX4 CGAX6 CGAX6M CGAX6MH CGAX8
CPE4 CPE4H CPE6M CPE6MH CPE8 CPE8H CPS4 CPS6M CPS8 CPS8R
CPEG6M CPEG6MH
CCL9 CCL12 CCL18 CCL24
MCL6 MCL9
R2D2 RAX2

Features tested

Automatic “shrink” fit

Small-sliding contact pair

Problem description

The models consist of elements with their contact surfaces initially overclosed. This initial overclosure is removed in the first step, which is nonlinear. The second step is a linear perturbation step, wherein relative sliding is performed between the two surfaces. The value of friction is changed in the third nonlinear step. The fourth step is a linear perturbation step, wherein relative frictional sliding is performed between the two surfaces. The fifth step is a direct-solution steady-state dynamic analysis of the two surfaces in contact. In the sixth step the natural frequencies are extracted, which are then used in the seventh step to conduct a subspace-based steady-state dynamic analysis.

A five-step test is carried out for generalized axisymmetric elements. Anisotropic friction is used throughout the test. The first and second steps are the same as mentioned earlier. The third step is a linear perturbation step, wherein relative twisting is performed between the two surfaces. Steps 4 and 5 are similar, except that Step 4 is a linear perturbation step. In these steps both relative sliding and twisting are performed between the two surfaces.

Only a four-step test is carried out for cylindrical-type elements.

Model:

Gap clearance	0.01
Gap clearance for cylindrical-type elements	0.2
Truss length	5.0
Truss area	0.5
2D solid element dimensions	$1.0 \times 5.0 \times 1.0$
Inner radius of axisymmetric solids and cylindrical-type elements	1.0

3D solid element dimensions	1.0 unit in each direction

Material:

Young's modulus	30×10^6
Poisson's ratio	0.3
Gap friction	0.0
Density	7700.0

Results and discussion

The contact pressure and tractions agree with the analytical results.

Input files

ei33sisc.inp	C3D4 elements.
ei33sisc_surf.inp	C3D4 elements, surface-to-surface constraint enforcement method.
ei34sisc.inp	C3D8 elements.
ei34sisc_surf.inp	C3D8 elements, surface-to-surface constraint enforcement method.
ei34sish.inp	C3D8H elements.
ei34sish_surf.inp	C3D8H elements, surface-to-surface constraint enforcement method.
ei35sisc.inp	C3D5 elements.
ei35sisc_surf.inp	C3D5 elements, surface-to-surface constraint enforcement method.
ei36sisc.inp	C3D10 elements.
ei36sisc_surf.inp	C3D10 elements, surface-to-surface constraint enforcement method.
ei36sisc_auglagr.inp	C3D10 elements, augmented Lagrangian contact model.
ei36sisc_auglagr_surf.inp	C3D10 elements, augmented Lagrangian contact model, surface-to-surface constraint enforcement method.
ei36sisc_c3d10hs.inp	C3D10HS elements.
ei36sisc_surf_c3d10hs.inp	C3D10HS elements, surface-to-surface constraint enforcement method.
ei36sisc_auglagr_c3d10hs.inp	C3D10HS elements, augmented Lagrangian contact model.
ei36sisc_auglagr_surf_c3d10hs.inp	C3D10HS elements, augmented Lagrangian contact model, surface-to-surface constraint enforcement method.
ei39sisc_c3d10m.inp	C3D10M elements.
ei39sisc_c3d10m_surf.inp	C3D10M elements, surface-to-surface constraint enforcement method.
ei39sisc_c3d10mh.inp	C3D10MH elements.
ei39sisc_c3d10mh_surf.inp	C3D10MH elements, surface-to-surface constraint enforcement method.
ei38sisc.inp	C3D20 elements.
ei38sisc_surf.inp	C3D20 elements, surface-to-surface constraint enforcement method.
ei38sish.inp	C3D20H elements.

ei38sish_surf.inp	C3D20H elements, surface-to-surface constraint enforcement method.
ei39sisc.inp	C3D27 elements.
ei39sisc_surf.inp	C3D27 elements, surface-to-surface constraint enforcement method.
eia2sisa.inp	CAX4 elements.
eia2sisa_surf.inp	CAX4 elements, surface-to-surface constraint enforcement method.
eia2sira.inp	CAX4, RAX2 elements.
eia2sira_surf.inp	CAX4, RAX2 elements, surface-to-surface constraint enforcement method.
eia2sish.inp	CAX4H elements.
eia2sish_surf.inp	CAX4H elements, surface-to-surface constraint enforcement method.
eia2sirh.inp	CAX4H, RAX2 elements.
eia2sirh_surf.inp	CAX4H, RAX2 elements, surface-to-surface constraint enforcement method.
eia3sisa_cax6m.inp	CAX6M elements.
eia3sisa_cax6m_surf.inp	CAX6M elements, surface-to-surface constraint enforcement method.
eia3sira_cax6m.inp	CAX6M, RAX2 elements.
eia3sira_cax6m_surf.inp	CAX6M, RAX2 elements, surface-to-surface constraint enforcement method.
eia3sisa_cax6mh.inp	CAX6MH elements.
eia3sisa_cax6mh_surf.inp	CAX6MH elements, surface-to-surface constraint enforcement method.
eia3sira_cax6mh.inp	CAX6MH, RAX2 elements.
eia2sisg3.inp	CGAX3 elements.
eia2sisg3_surf.inp	CGAX3 elements, surface-to-surface constraint enforcement method.
eia2sirg3.inp	CGAX3, RAX2 elements.
eia2sirg3_surf.inp	CGAX3, RAX2 elements, surface-to-surface constraint enforcement method.
eia2sisg4.inp	CGAX4 elements.
eia2sisg4_surf.inp	CGAX4 elements, surface-to-surface constraint enforcement method.
eia2sirg4.inp	CGAX4, RAX2 elements.
eia2sirg4_surf.inp	CGAX4, RAX2 elements, surface-to-surface constraint enforcement method.
eia3sisg6.inp	CGAX6 elements.
eia3sisg6_surf.inp	CGAX6 elements, surface-to-surface constraint enforcement method.
eia3sirg6.inp	CGAX6, RAX2 elements.
eia3sirg6_surf.inp	CGAX6, RAX2 elements, surface-to-surface constraint enforcement method.
eia3sisg6_cgax6m.inp	CGAX6M elements.
eia3sisg6_cgax6m_surf.inp	CGAX6M elements, surface-to-surface constraint enforcement method.
eia3sirg6_cgax6m.inp	CGAX6M, RAX2 elements.
eia3sirg6_cgax6m_surf.inp	CGAX6M, RAX2 elements, surface-to-surface constraint enforcement method.
eia3sisg6_cgax6mh.inp	CGAX6MH elements.

eia3sisg6_cgax6mh_surf.inp	CGAX6MH elements, surface-to-surface constraint enforcement method.
eia3sirg6_cgax6mh.inp	CGAX6MH, RAX2 elements.
eia3sirg6_cgax6mh_surf.inp	CGAX6MH, RAX2 elements, surface-to-surface constraint enforcement method.
eia3sisg8.inp	CGAX8 elements.
eia3sisg8_surf.inp	CGAX8 elements, surface-to-surface constraint enforcement method.
eia3sirg8.inp	CGAX8, RAX2 elements.
eia3sirg8_surf.inp	CGAX8, RAX2 elements, surface-to-surface constraint enforcement method.
ei22sise.inp	CPE4 elements.
ei22sise_surf.inp	CPE4 elements, surface-to-surface constraint enforcement method.
ei22sire.inp	CPE4, R2D2 elements.
ei22sire_surf.inp	CPE4, R2D2 elements, surface-to-surface constraint enforcement method.
ei22sish.inp	CPE4H elements.
ei22sish_surf.inp	CPE4H elements, surface-to-surface constraint enforcement method.
ei22sirh.inp	CPE4H, R2D2 elements.
ei22sirh_surf.inp	CPE4H, R2D2 elements, surface-to-surface constraint enforcement method.
ei23sise_cpe6m.inp	CPE6M elements.
ei23sise_cpe6m_surf.inp	CPE6M elements, surface-to-surface constraint enforcement method.
ei23sise_cpe6m_auglagr.inp	CPE6M elements, augmented Lagrangian contact model.
ei23sise_cpe6m_auglagr_surf.inp	CPE6M elements, augmented Lagrangian contact model, surface-to-surface constraint enforcement method.
ei23sire_cpe6m.inp	CPE6M, R2D2 elements.
ei23sire_cpe6m_surf.inp	CPE6M, R2D2 elements, surface-to-surface constraint enforcement method.
ei23sise_cpe6mh.inp	CPE6MH elements.
ei23sise_cpe6mh_surf.inp	CPE6MH elements, surface-to-surface constraint enforcement method.
ei23sire_cpe6mh.inp	CPE6MH, R2D2 elements.
ei23sire_cpe6mh_surf.inp	CPE6MH, R2D2 elements, surface-to-surface constraint enforcement method.
ei23sise.inp	CPE8 elements.
ei23sise_surf.inp	CPE8 elements, surface-to-surface constraint enforcement method.
ei23sise_auglagr.inp	CPE8 elements.
ei23sise_auglagr_surf.inp	CPE8 elements, surface-to-surface constraint enforcement method.
ei23sire.inp	CPE8, R2D2 elements.
ei23sire_surf.inp	CPE8, R2D2 elements, surface-to-surface constraint enforcement method.
ei23sire_auglagr.inp	CPE8, R2D2 elements.
ei23sire_auglagr_surf.inp	CPE8, R2D2 elements, surface-to-surface constraint enforcement method.
ei23sireimport_auglagr.inp	CPE8, R2D2 elements, import analysis.

ei23sireimport_auglagr_surf.inp	CPE8, R2D2 elements, import analysis, surface-to-surface constraint enforcement method.
ei23sish.inp	CPE8H elements.
ei23sish_surf.inp	CPE8H elements, surface-to-surface constraint enforcement method.
ei23sirh.inp	CPE8H, R2D2 elements.
ei23sirh_surf.inp	CPE8H, R2D2 elements, surface-to-surface constraint enforcement method.
ei22siss.inp	CPS4 elements.
ei22siss_surf.inp	CPS4 elements,surface-to-surface constraint enforcement method.
ei22sirs.inp	CPS4, R2D2 elements.
ei22sirs_surf.inp	CPS4, R2D2 elements, surface-to-surface constraint enforcement method.
ei23siss_cps6m.inp	CPS6M elements.
ei23siss_cps6m_surf.inp	CPS6M elements, surface-to-surface constraint enforcement method.
ei23sirs_cps6m.inp	CPS6M, R2D2 elements.
ei23sirs_cps6m_surf.inp	CPS6M, R2D2 elements, surface-to-surface constraint enforcement method.
ei22siss.inp	CPS8 elements.
ei22siss_surf.inp	CPS8 elements,surface-to-surface constraint enforcement method.
ei23sirs.inp	CPS8, R2D2 elements.
ei23sirs_surf.inp	CPS8, R2D2 elements, surface-to-surface constraint enforcement method.
ei23sise_cpeg6m.inp	CPEG6M elements.
ei23sise_cpeg6m_surf.inp	CPEG6M elements, surface-to-surface constraint enforcement method.
ei23sire_cpeg6m.inp	CPEG6M, R2D2 elements.
ei23sire_cpeg6m_surf.inp	CPEG6M, R2D2 elements, surface-to-surface constraint enforcement method.
ei23sise_cpeg6mh.inp	CPEG6MH elements.
ei23sise_cpeg6mh_surf.inp	CPEG6MH elements, surface-to-surface constraint enforcement method.
ei36sisc_ccl9.inp	CCL9, MCL9 elements.
ei36sisc_ccl9_thick.inp	CCL9, MCL9 elements, thickness is considered.
ei36sisc_ccl9_surf.inp	CCL9, MCL9 elements, surface-to-surface constraint enforcement method.
ei36sisc_ccl9_surf_nothick.inp	CCL9, MCL9 elements, surface-to-surface constraint enforcement method, surface thickness effects ignored.
ei36sirc_ccl9.inp	CCL9 elements.
ei36sirc_ccl9_surf.inp	CCL9 elements, surface-to-surface constraint enforcement method.
ei36sisc_ccl12.inp	CCL12, MCL9 elements.
ei36sisc_ccl12_thick.inp	CCL12, MCL9 elements, thickness is considered.
ei36sisc_ccl12_surf.inp	CCL12, MCL9 elements, surface-to-surface constraint enforcement method.
ei36sirc_ccl12.inp	CCL12 elements.
ei36sirc_ccl12_surf.inp	CCL12 elements, surface-to-surface constraint enforcement method.

ei39sisc_ccl18.inp	CCL18, MCL9 elements.
ei39sisc_ccl18_surf.inp	CCL18, MCL9 elements, surface-to-surface constraint enforcement method.
ei39sirc_ccl18.inp	CCL18 elements.
ei39sirc_ccl18_surf.inp	CCL18 elements, surface-to-surface constraint enforcement method.
ei39sisc_ccl24.inp	CCL24, MCL9 elements.
ei39sisc_ccl24_surf.inp	CCL24, MCL9 elements, surface-to-surface constraint enforcement method.
ei39sirc_ccl24.inp	CCL24 elements.
ei39sirc_ccl24_surf.inp	CCL24 elements, surface-to-surface constraint enforcement method.
ei36sisc_mcl6.inp	MCL6, MCL9 elements.
ei36sirc_mcl6.inp	MCL6 elements.
ei36sirc_mcl6_surf.inp	MCL6 elements, surface-to-surface constraint enforcement method.
ei39sisc_mcl9.inp	MCL9 elements.
ei39sisc_mcl9_surf.inp	MCL9 elements, surface-to-surface constraint enforcement method.
ei39sirc_mcl9.inp	MCL9 elements.
ei39sirc_mcl9_surf.inp	MCL9 elements, surface-to-surface constraint enforcement method.

Small-sliding contact between coupled temperature-displacement surfaces

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D4T C3D6T C3D8IT C3D8HT C3D8RT C3D8RHT C3D8T C3D10MT C3D10MHT C3D20HT C3D20T
CAX3T CAX4HT CAX4RT CAX4RHT CAX4T CAX6MT CAX6MHT CAX8HT CAX8T
CGAX4HT CGAX4RT CGAX4RHT CGAX4T CGAX6MT CGAX6MHT CGAX8HT CGAX8T

CPE3T CPE4HT CPE4RT CPE4RHT CPE4T CPE6MT CPE6MHT CPE8HT CPE8T
CPEG3HT CPEG3T CPEG4HT CPEG4RT CPEG4RHT CPEG4T CPEG6MT CPEG6MHT CPEG8HT CPEG8RHT CPEG8T
CPS3T CPS4RT CPS4T CPS6MT CPS8T
S3RT S3T S4RT S4T S8RT SAX2T SC6RT SC8RT

Features tested

Gap conductance

Gap heat generation

Gap radiation

Small-sliding contact pair

Problem description

The models consist of two bodies lying next to each other. Heat transfer across the gap between the two bodies can take place via gap conductance, gap heat generation, or gap radiation. Only heat transfer via gap conductance and gap radiation is tested for the shells.

For the continuum elements we initiate heat flow in the first step by applying different constant temperature fields to each solid body. The steady-state temperature along both sides of the interface is used to verify the numerical solutions. The gap closes due to thermal expansion of the two bodies. In the second step the top block is displaced relative to the bottom block to generate heat due to frictional sliding. In addition, heat transfer occurs due to gap conductance and gap radiation. The upper body is displaced back to its original position in the third step. In Abaqus/Standard a fourth step is also included. This step is a linear perturbation step, wherein a load of sufficient magnitude to open the gap is applied. In addition, in Abaqus/Standard the tied contact pair formulation is verified by defining one of the deformable bodies with this feature. The section output requests to the results (`.fil`) file and to the data (`.dat`) file are used to output the total force and the total heat flux across the contact surfaces; the results match the output of similar output quantities obtained when writing contact variables for surface contact pairs to the results file.

For the shells, heat flow is initiated by applying different constant temperatures to degree of freedom 15 of the top body and to degree of freedom 11 of the bottom body. The steady-state temperature along both sides of the interface is used to verify the numerical solutions.

Material:

Young's modulus	30×10^6
Poisson's ratio	0.3
Gap friction	0.01
Density	7700.
Thermal expansion coeff.	10×10^{-6}
Conductivity	43.0
Specific heat	600.
Gap conductance	1.0
F_A	1×10^{-9}
F_B	1×10^{-9}
f_g	0.5

Results and discussion

The finite element results agree with the analytical results.

Input files

Abaqus/Standard input files

coupledtempsmslcont_s_c3d4t.inp	C3D4T elements.
coupledtempsmslcont_s_c3d4t_surf.inp	C3D4T elements, surface-to-surface constraint enforcement method.
coupledtempsmslcont_s_c3d6t.inp	C3D6T elements.
coupledtempsmslcont_s_c3d6t_surf.inp	C3D6T elements, surface-to-surface constraint enforcement method.
coupledtempsmslcont_s_c3d8it.inp	C3D8IT elements.
coupledtempsmslcont_s_c3d8it_surf.inp	C3D8IT elements, surface-to-surface constraint enforcement method.
ei34tish.inp	C3D8HT elements.
ei34tish_surf.inp	C3D8HT elements, surface-to-surface constraint enforcement method.
ei34tisr.inp	C3D8RT elements.
ei34tisr_surf.inp	C3D8RT elements, surface-to-surface constraint enforcement method.
ei34tisy.inp	C3D8RHT elements.
ei34tisy_surf.inp	C3D8RHT elements, surface-to-surface constraint enforcement method.
ei34tisc.inp	C3D8T elements.
ei34tisc_surf.inp	C3D8T elements, surface-to-surface constraint enforcement method.
ei36tisc.inp	C3D10MT elements.
ei36tisc_surf.inp	C3D10MT elements, surface-to-surface constraint enforcement method.

ei36tish.inp	C3D10MHT elements.
ei36tish_surf.inp	C3D10MHT elements, surface-to-surface constraint enforcement method.
ei38tish.inp	C3D20HT elements.
ei38tish_surf.inp	C3D20HT elements, surface-to-surface constraint enforcement method.
ei38tisc.inp	C3D20T elements.
ei38tisc_surf.inp	C3D20T elements, surface-to-surface constraint enforcement method.
ei38tisc_auglagr.inp	C3D20T elements, augmented Lagrangian contact model.
ei38tisc_auglagr_surf.inp	C3D20T elements, augmented Lagrangian contact model, surface-to-surface constraint enforcement method.
coupledtempsmslcont_s_cax3t.inp	CAX3T elements.
coupledtempsmslcont_s_cax3t_surf.inp	CAX3T elements, surface-to-surface constraint enforcement method.
eia2tish.inp	CAX4HT elements.
eia2tish_surf.inp	CAX4HT elements, surface-to-surface constraint enforcement method.
eia2tisr.inp	CAX4RT elements.
eia2tisr_surf.inp	CAX4RT elements, surface-to-surface constraint enforcement method.
eia2tisy.inp	CAX4RHT elements.
eia2tisy_surf.inp	CAX4RHT elements, surface-to-surface constraint enforcement method.
eia2tisa.inp	CAX4T elements.
eia2tisa_surf.inp	CAX4T elements, surface-to-surface constraint enforcement method.
eia3tisa_cax6mt.inp	CAX6MT elements.
eia3tisa_cax6mt_surf.inp	CAX6MT elements, surface-to-surface constraint enforcement method.
eia3tisa_cax6mht.inp	CAX6MHT elements.
eia3tisa_cax6mht_surf.inp	CAX6MHT elements, surface-to-surface constraint enforcement method.
eia3tish.inp	CAX8HT elements.
eia3tish_surf.inp	CAX8HT elements, surface-to-surface constraint enforcement method.
eia3tisa.inp	CAX8T elements.
eia3tisa_surf.inp	CAX8T elements, surface-to-surface constraint enforcement method.
eig2tish.inp	CGAX4HT elements.
eig2tish_surf.inp	CGAX4HT elements, surface-to-surface constraint enforcement method.
eig2tisr.inp	CGAX4RT elements.
eig2tisr_surf.inp	CGAX4RT elements, surface-to-surface constraint enforcement method.
eig2tisy.inp	CGAX4RHT elements.
eig2tisy_surf.inp	CGAX4RHT elements, surface-to-surface constraint enforcement method.
eig2tisa.inp	CGAX4T elements.

eig2tisa_surf.inp	CGAX4T elements, surface-to-surface constraint enforcement method.
eig3tisa_cgax6mt.inp	CGAX6MT elements.
eig3tisa_cgax6mt_surf.inp	CGAX6MT elements, surface-to-surface constraint enforcement method.
eig3tisa_cgax6mht.inp	CGAX6MHT elements.
eig3tisa_cgax6mht_surf.inp	CGAX6MHT elements, surface-to-surface constraint enforcement method.
eig3tish.inp	CGAX8HT elements.
eig3tish_surf.inp	CGAX8HT elements, surface-to-surface constraint enforcement method.
eig3tisa.inp	CGAX8T elements.
eig3tisa_surf.inp	CGAX8T elements, surface-to-surface constraint enforcement method.
ei22tish.inp	CPE4HT elements.
ei22tish_surf.inp	CPE4HT elements, surface-to-surface constraint enforcement method.
ei22tise_cpe4rt.inp	CPE4RT elements.
ei22tise_cpe4rt_surf.inp	CPE4RT elements, surface-to-surface constraint enforcement method.
ei22tise_cpe4rht.inp	CPE4RHT elements.
ei22tise_cpe4rht_surf.inp	CPE4RHT elements, surface-to-surface constraint enforcement method.
coupledtempsmslcont_s_cpe3t.inp	CPE3T elements.
coupledtempsmslcont_s_cpe3t_surf.inp	CPE3T elements, surface-to-surface constraint enforcement method.
ei22tise.inp	CPE4T elements.
ei22tise_surf.inp	CPE4T elements, surface-to-surface constraint enforcement method.
ei23tise_cpe6mt.inp	CPE6MT elements.
ei23tise_cpe6mt_surf.inp	CPE6MT elements, surface-to-surface constraint enforcement method.
ei23tise_cpe6mht.inp	CPE6MHT elements.
ei23tise_cpe6mht_surf.inp	CPE6MHT elements, surface-to-surface constraint enforcement method.
ei23tish.inp	CPE8HT elements.
ei23tish_surf.inp	CPE8HT elements, surface-to-surface constraint enforcement method.
ei23tise.inp	CPE8T elements.
ei23tise_surf.inp	CPE8T elements, surface-to-surface constraint enforcement method.
ei23tise_auglagr.inp	CPE8T elements.
ei23tise_auglagr_surf.inp	CPE8T elements, surface-to-surface constraint enforcement method.
eit2tish.inp	CPEG3HT elements.
eit2tish_surf.inp	CPEG3HT elements, surface-to-surface constraint enforcement method.
eit2tise.inp	CPEG3T elements.
eit2tise_surf.inp	CPEG3T elements, surface-to-surface constraint enforcement method.

eip2tish.inp	CPEG4HT elements.
eip2tish_surf.inp	CPEG4HT elements, surface-to-surface constraint enforcement method.
eip2tisr.inp	CPEG4RT elements.
eip2tisr_surf.inp	CPEG4RT elements, surface-to-surface constraint enforcement method.
eip2tisx.inp	CPEG4RHT elements.
eip2tisx_surf.inp	CPEG4RHT elements, surface-to-surface constraint enforcement method.
eip2tise.inp	CPEG4T elements.
eip2tise_surf.inp	CPEG4T elements, surface-to-surface constraint enforcement method.
eip3tise_cpeg6mt.inp	CPEG6MT elements.
eip3tise_cpeg6mt_surf.inp	CPEG6MT elements, surface-to-surface constraint enforcement method.
eip3tise_cpeg6mht.inp	CPEG6MHT elements.
eip3tise_cpeg6mht_surf.inp	CPEG6MHT elements, surface-to-surface constraint enforcement method.
eip3tish.inp	CPEG8HT elements.
eip3tish_surf.inp	CPEG8HT elements, surface-to-surface constraint enforcement method.
eip3tisx.inp	CPEG8RHT elements.
eip3tisx_surf.inp	CPEG8RHT elements, surface-to-surface constraint enforcement method.
eip3tise.inp	CPEG8T elements.
eip3tise_surf.inp	CPEG8T elements, surface-to-surface constraint enforcement method.
coupledtempsmslcont_s_cps3t.inp	CPS3T elements.
coupledtempsmslcont_s_cps3t_surf.inp	CPS3T elements, surface-to-surface constraint enforcement method.
ei22tisr.inp	CPS4RT elements.
ei22tisr_surf.inp	CPS4RT elements, surface-to-surface constraint enforcement method.
ei22tiss.inp	CPS4T elements.
ei22tiss_surf.inp	CPS4T elements, surface-to-surface constraint enforcement method.
ei23tise_cps6mt.inp	CPS6MT elements.
ei23tise_cps6mt_surf.inp	CPS6MT elements, surface-to-surface constraint enforcement method.
ei23tiss.inp	CPS8T elements.
ei23tiss_surf.inp	CPS8T elements, surface-to-surface constraint enforcement method.
ei23tiss_auglagr.inp	CPS8T elements, augmented Lagrangian contact model.
ei23tiss_auglagr_surf.inp	CPS8T elements, augmented Lagrangian contact model, surface-to-surface constraint enforcement method.
coupledtempsmslcont_s_s3rt.inp	S3RT elements.
coupledtempsmslcont_s_s3rt_surf.inp	S3RT elements, surface-to-surface constraint enforcement method.
coupledtempsmslcont_s_s3t.inp	S3T elements.

coupledtempsmslcont_s_s3t_surf.inp	S3T elements, surface-to-surface constraint enforcement method.
coupledtempsmslcont_s_s4rt.inp	S4RT elements.
coupledtempsmslcont_s_s4rt_surf.inp	S4RT elements, surface-to-surface constraint enforcement method.
coupledtempsmslcont_s_s4t.inp	S4T elements.
coupledtempsmslcont_s_s4t_surf.inp	S4T elements, surface-to-surface constraint enforcement method.
ei38tiss.inp	S8RT elements.
ei38tiss_surf.inp	S8RT elements, surface-to-surface constraint enforcement method.
ei38tissp.inp	Postprocessing to recover additional output from the restart file of the analysis testing S8RT elements.
ei38tissp_surf.inp	Postprocessing to recover additional output from the restart file of the analysis testing S8RT elements, surface-to-surface constraint enforcement method.
eia3tiss.inp	SAX2T elements.
eia3tiss_surf.inp	SAX2T elements, surface-to-surface constraint enforcement method.
eia3tiss_surf_nothick.inp	SAX2T elements, surface-to-surface constraint enforcement method, surface thickness effects ignored.
eiu1tgc1.inp	T3D2T elements with GAPUNIT elements.
eiu1tgc1_surf.inp	T3D2T elements with GAPUNIT elements, surface-to-surface constraint enforcement method.
coupledtempsmslcont_s_sc6rt.inp	SC6RT elements.
coupledtempsmslcont_s_sc6rt_surf.inp	SC6RT elements, surface-to-surface constraint enforcement method.
coupledtempsmslcont_s_sc8rt.inp	SC8RT elements.
coupledtempsmslcont_s_sc8rt_surf.inp	SC8RT elements, surface-to-surface constraint enforcement method.

Abaqus/Explicit input files

coupledtempsmslcont_x_c3d4t.inp	C3D4T elements.
coupledtempsmslcont_x_c3d6t.inp	C3D6T elements.
coupledtempsmslcont_x_c3d8rt.inp	C3D8RT elements.
coupledtempsmslcont_x_c3d8t.inp	C3D8T elements.
coupledtempsmslcont_x_c3d10mt.inp	C3D10MT elements.
coupledtempsmslcont_x_sc8rt.inp	SC8RT elements.
coupledtempsmslcont_x_cax3t.inp	CAX3T elements.
coupledtempsmslcont_x_cax4rt.inp	CAX4RT elements.
coupledtempsmslcont_x_cax6mt.inp	CAX6MT elements.
coupledtempsmslcont_x_cpe3t.inp	CPE3T elements.
coupledtempsmslcont_x_cpe4rt.inp	CPE4RT elements.
coupledtempsmslcont_x_cpe6mt.inp	CPE6MT elements.
coupledtempsmslcont_x_cps3t.inp	CPS3T elements.
coupledtempsmslcont_x_cps4rt.inp	CPS4RT elements.
coupledtempsmslcont_x_cps6mt.inp	CPS6MT elements.

Small-sliding contact between coupled thermal-electrical-structural surfaces

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

Q3D4 Q3D6 Q3D8 Q3D8H Q3D8R Q3D8RH Q3D10M Q3D10MH Q3D20 Q3D20H Q3D20R Q3D20RH QAX3 QAX4 QAX6M QAX8

Features tested

Gap conductance

Gap electrical conductance

Gap heat generation

Gap radiation

Small-sliding contact pair

Problem description

The models consist of two bodies lying next to each other. Heat transfer and current flow across the gap between the two bodies can take place via gap conductance, gap heat generation, gap radiation, or gap electrical conductance.

We initiate heat and current flow in the first step by applying different constant temperature and electrical potential fields to each solid body. Heat transfer occurs due to gap conductance and gap radiation; and the current flow occurs due to gap electrical conductance. The steady-state fields along both sides of the interface are used to verify the numerical solutions. The gap closes due to thermal expansion of the two bodies. In the second step the top block is displaced relative to the bottom block. The upper body is displaced back to its original position in the third step. The fourth step is a linear perturbation step, wherein a load of sufficient magnitude to open the gap is applied. In addition, the tied contact pair formulation is verified by defining one of the deformable bodies with this feature. The section output requests to the results (`.fil`) file and to the data (`.dat`) file are used to output the total force and the total heat and current fluxes across the contact surfaces; the results match the output of similar output quantities obtained when writing contact variables for surface contact pairs to the results file.

The mesh in *Figure 1*, used for axisymmetric tests, is representative of all meshes used in these tests.

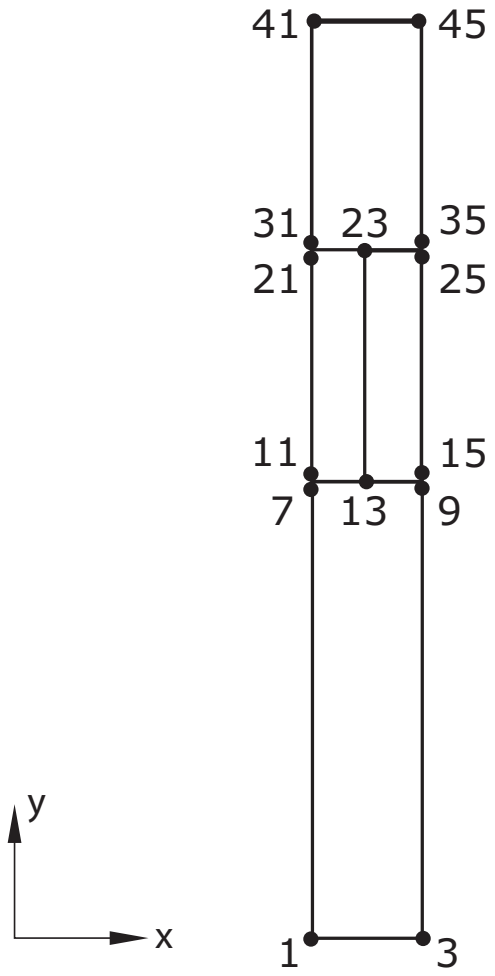


Figure 1: Representative mesh.

Material:

Young's modulus	30×10^6
Poisson's ratio	0.3
Gap friction	0.01
Density	7700.
Thermal expansion coefficient	10×10^{-6}
Conductivity	43.0
Electrical conductivity	43.0
Joule heat fraction	0.0
Specific heat	600.
Gap conductance	1.0
ϵ_A	0.034664
ϵ_B	0.034664
F	1.0

θ^z	–273.16
η	0.5
f	0.5
Gap electrical conductance	1.0

Results and discussion

The finite element results agree with the analytical results.

Input files

Abaqus/Standard input files

tes_smslcont_q3d4.inp	Q3D4 elements.
tes_smslcont_q3d4_surf.inp	Q3D4 elements, surface-to-surface constraint enforcement method.
tes_smslcont_q3d6.inp	Q3D6 elements.
tes_smslcont_q3d6_surf.inp	Q3D6 elements, surface-to-surface constraint enforcement method.
tes_smslcont_q3d8.inp	Q3D8 elements.
tes_smslcont_q3d8_er.inp	Q3D8 elements, with **ELECTRICAL RESISTIVITY* insteady of **ELECTRICAL CONDUCTIVITY*.
tes_smslcont_q3d8_surf.inp	Q3D8 elements, surface-to-surface constraint enforcement method.
tes_smslcont_q3d8h.inp	Q3D8H elements.
tes_smslcont_q3d8h_surf.inp	Q3D8H elements, surface-to-surface constraint enforcement method.
tes_smslcont_q3d8r.inp	Q3D8R elements.
tes_smslcont_q3d8r_surf.inp	Q3D8R elements, surface-to-surface constraint enforcement method.
tes_smslcont_q3d8rh.inp	Q3D8RH elements.
tes_smslcont_q3d8rh_surf.inp	Q3D8RH elements, surface-to-surface constraint enforcement method.
tes_smslcont_q3d10m.inp	Q3D10M elements.
tes_smslcont_q3d10m_surf.inp	Q3D10M elements, surface-to-surface constraint enforcement method.
tes_smslcont_q3d10mh.inp	Q3D10MH elements.
tes_smslcont_q3d10mh_surf.inp	Q3D10MH elements, surface-to-surface constraint enforcement method.
tes_smslcont_q3d20.inp	Q3D20 elements.
tes_smslcont_q3d20_surf.inp	Q3D20 elements, surface-to-surface constraint enforcement method.
tes_smslcont_q3d20_auglagr.inp	Q3D20 elements, augmented Lagrangian contact model.

tes_smslcont_q3d20_auglagr_surf.inp	Q3D20 elements, augmented Lagrangian contact model, surface-to-surface constraint enforcement method.
tes_smslcont_q3d20h.inp	Q3D20H elements.
tes_smslcont_q3d20h_surf.inp	Q3D20H elements, surface-to-surface constraint enforcement method.
tes_smslcont_q3d20r.inp	Q3D20R elements.
tes_smslcont_q3d20r_surf.inp	Q3D20R elements, surface-to-surface constraint enforcement method.
tes_smslcont_q3d20rh.inp	Q3D20RH elements.
tes_smslcont_q3d20rh_surf.inp	Q3D20RH elements, surface-to-surface constraint enforcement method.
tes_smslcont_qax3.inp	QAX3 elements.
tes_smslcont_qax3_surf.inp	QAX3 elements, surface-to-surface constraint enforcement method.
tes_smslcont_qax4.inp	QAX4 elements.
tes_smslcont_qax4_er.inp	QAX4 elements, with **ELECTRICAL RESISTIVITY* insteady of **ELECTRICAL CONDUCTIVITY*.
tes_smslcont_qax4_surf.inp	QAX4 elements, surface-to-surface constraint enforcement method.
tes_smslcont_qax6m.inp	QAX6M elements.
tes_smslcont_qax6m_surf.inp	QAX6M elements, surface-to-surface constraint enforcement method.
tes_smslcont_qax8.inp	QAX8 elements.
tes_smslcont_qax8_surf.inp	QAX8 elements, surface-to-surface constraint enforcement method.

Small-sliding contact between coupled pore pressure-displacement elements

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

CAX8P CPE8P

Features tested

Small-sliding contact pair

Problem description

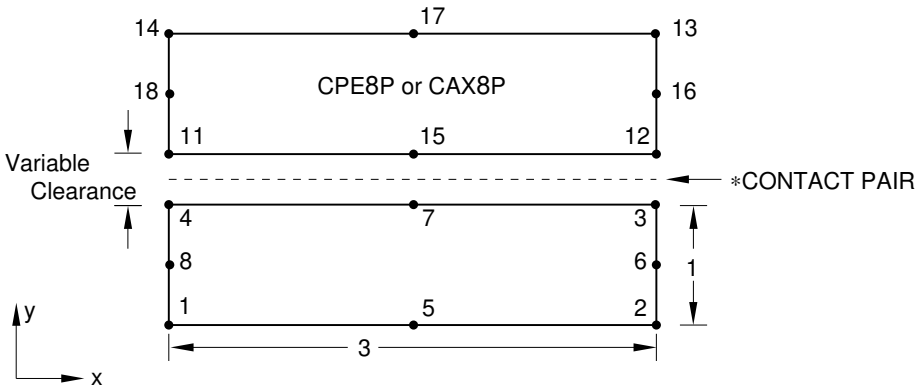


Figure 1: Element topology for all tests.

Model: The out-of-plane thickness for all elements is 0.5.

Material: The elastic properties of the soil are Young's modulus = 1×10^8 and Poisson's ratio = 0.0. The permeability of the soil = 1×10^{-4}. The initial void ratio = 1.0 for all tests.

Boundary conditions: In all tests, nodes are restrained in the 1-direction.

Analysis tests

There are four different tests.

Initial conditions test

This test verifies that the initial pore fluid pressure procedure works with a contact pair. All nodes in the model are initialized to a pore pressure of −50.0.

Consolidation test

The consolidation test verifies that the contact pair procedure works properly with the transient soils consolidation procedure. The test is essentially a one-dimensional problem where two surfaces are brought together at a constant rate, as shown in *Figure 2*.

Point A in the figure corresponds to nodes 1, 5, and 2; point B corresponds to nodes 4, 7, and 3; and so on. As points C and B move toward each other, fluid rushes out through points A and D. This gives rise to a compressive stress state in the soil segments AB and CD. A pore pressure field develops to balance out the effective stresses.

Steady-state test

The steady-state test verifies that the contact pair procedure works properly with soil consolidation procedure. The problem is the same one that is modeled in the consolidation test. There is zero stress and zero pore pressure at steady state; therefore, it is necessary to use solution controls to avoid convergence difficulties as a result of the fact that both the time average force and the force residuals are practically zero.

Interference test

The interference test verifies that a combination of interface overclosure and pore pressure gradient is handled correctly by the contact pair procedure. The test is essentially a one-dimensional problem where two surfaces start with an interference fit and a pore pressure gradient exists across the two bodies. The steady-state equilibrium is sought.

Results and discussion

Most of the input files used for these tests employ an unsymmetric equation solver. Using the unsymmetric solver improves convergence in steady-state analyses.

Initial conditions test

The pore pressure at every node should be −50.0.

Consolidation test

From Darcy's law we find that during the first step of the analysis the effective stress profile is as shown in *Figure 4*.

From equilibrium of tractions we find that the pore pressure distribution is as shown in *Figure 5*. After the surfaces have stopped moving toward each other, the stresses and pore pressure quickly drop to zero. This is modeled in the second step of the analysis.

Steady-state test

The steady-state result is zero stress and zero pore pressure.

Interference test

This problem can be analyzed as a linear superposition of two states, as shown in *Figure 6*.

Input files

ei13psi1.inp	Initial conditions test, CPE8P elements.
ei13psi1_surf.inp	Initial conditions test, CPE8P elements, surface-to-surface constraint enforcement method.

ei13psi1_auglagr.inp	Initial conditions test, CPE8P elements.
ei13psi1_auglagr_surf.inp	Initial conditions test, CPE8P elements, surface-to-surface constraint enforcement method.
eia3psi1.inp	Initial conditions test, CAX8P elements.
eia3psi1_surf.inp	Initial conditions test, CAX8P elements, surface-to-surface constraint enforcement method.
ei13psnc.inp	Consolidation test, CPE8P elements.
ei13psnc_surf.inp	Consolidation test, CPE8P elements, surface-to-surface constraint enforcement method.
ei13psnc_auglagr.inp	Consolidation test, CPE8P elements.
ei13psnc_auglagr_surf.inp	Consolidation test, CPE8P elements, surface-to-surface constraint enforcement method.
eia3psnc.inp	Consolidation test, CAX8P elements.
eia3psnc_surf.inp	Consolidation test, CAX8P elements, surface-to-surface constraint enforcement method.
ei13psns.inp	Steady-state test, CPE8P elements.
ei13psns_surf.inp	Steady-state test, CPE8P elements, surface-to-surface constraint enforcement method.
ei13psns.inp	Steady-state test, CPE8P elements.
ei13psns_surf.inp	Steady-state test, CPE8P elements, surface-to-surface constraint enforcement method.
ei13psni.inp	Interference test, CPE8P elements.
ei13psni_surf.inp	Interference test, CPE8P elements, surface-to-surface constraint enforcement method.
ei13psni_auglagr.inp	Interference test, CPE8P elements.
ei13psni_auglagr_surf.inp	Interference test, CPE8P elements, surface-to-surface constraint enforcement method.
eia3psni.inp	Interference test, CAX8P elements.
eia3psni_surf.inp	Interference test, CAX8P elements, surface-to-surface constraint enforcement method.

Figures

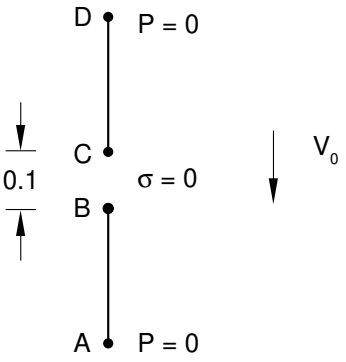


Figure 2: One-dimensional consolidation test.

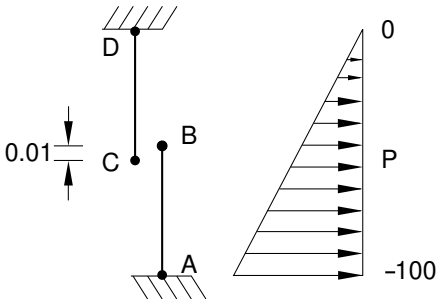


Figure 3: Interference test.

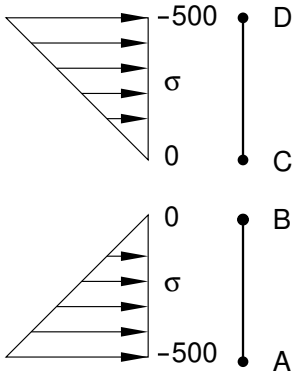


Figure 4: Effective stress profile for the first step of the consolidation test.

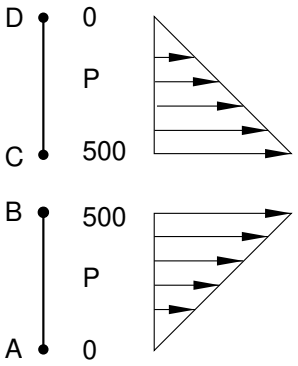


Figure 5: Pore pressure distribution for the consolidation test.

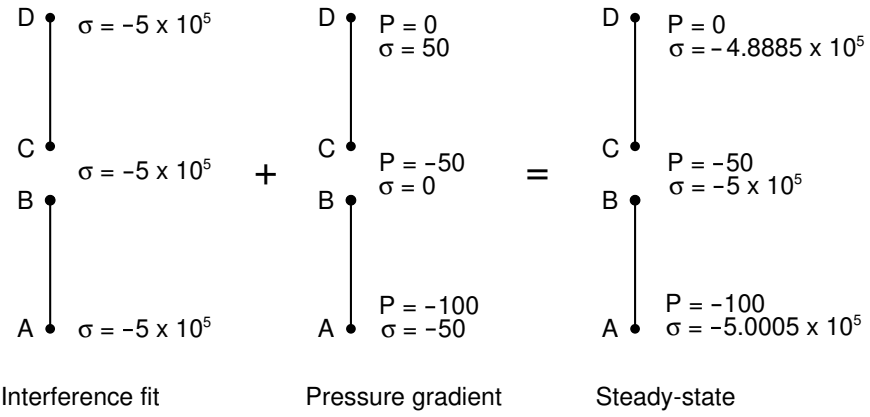


Figure 6: Linear superposition of two states used to solve the interference test problem.

Finite-sliding contact between stress/displacement elements

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D4 C3D5 C3D8 C3D8H C3D10M C3D20
CAX4 CAX8 CGAX3 CGAX4 CGAX6 CGAX6M CGAX8
CPE4 CPE6M CPE6MH CPE8 CPS6M
CCL12 CCL24
MCL6 MCL9
ITT21 ITT31

Features tested

Contact pair

Problem description

This section deals with the surface-based approach to contact between stress/displacement elements.

Most tests are run with and without friction. A coefficient of friction of 0.2 is used in all tests with isotropic friction. In most tests Step 1 results in contact and Step 2 initiates sliding. Several tests conduct direct-integration and subspace-based steady-state dynamic analyses.

Results and discussion

The contact pressure and tractions agree with the analytical results.

Input files

Zero friction:

ei304fcz.inp	C3D4 elements.
ei304fcz_surf.inp	C3D4 elements using surface-to-surface contact.
ei305fcz.inp	C3D5 elements.
ei305fcz_surf.inp	C3D5 elements using surface-to-surface contact.
ei308fhz.inp	C3D8/C3D8H elements.
ei308fhz_surf.inp	C3D8/C3D8H elements using surface-to-surface contact.
eig08fcz.inp	C3D8 elements, node-based surface.
ei310fmz.inp	C3D10M elements.
ei310fmz_surf.inp	C3D10M elements using surface-to-surface contact.
ei320fcz.inp	C3D20 elements.
ei320fcz_surf.inp	C3D20 elements using surface-to-surface contact.
eia04faz.inp	CAX4 elements.

eia04faz_surf.inp	CAX4 elements using surface-to-surface contact.
eia08faz.inp	CAX8
eia08faz_surf.inp	CAX8 elements using surface-to-surface contact
ei204fez.inp	CPE4 elements.
ei204fez_surf.inp	CPE4 elements using surface-to-surface contact.
ei206fmz.inp	CPE6M elements.
ei206fmz_surf.inp	CPE6M elements using surface-to-surface contact.
ei206fhz.inp	CPE6MH elements.
ei208fez.inp	CPE8 elements.
ei208fez_surf.inp	CPE8 elements using surface-to-surface contact.
ei206fsz.inp	CPS6M elements.
ei206fsz_surf.inp	CPS6M elements using surface-to-surface contact.

Friction:

ei304fcf.inp	C3D4 elements.
ei304fcf_surf.inp	C3D4 elements using surface-to-surface contact.
ei305fcf.inp	C3D5 elements.
ei305fcf_surf.inp	C3D5 elements using surface-to-surface contact.
ei308fcf.inp	C3D8 elements.
ei308fcf_surf.inp	C3D8 elements using surface-to-surface contact.
ei308fhf.inp	C3D8/C3D8H elements.
ei308fhf_surf.inp	C3D8/C3D8H elements using surface-to-surface contact.
eig08fcf.inp	C3D8 elements, node-based surface.
ei310fmf.inp	C3D10M elements.
ei310fmf_surf.inp	C3D10M elements using surface-to-surface contact.
ei320fcf.inp	C3D20 elements.
ei320fcf_surf.inp	C3D20 elements using surface-to-surface contact.
ei320fcf_auglagr.inp	C3D20 elements.
ei320fcf_auglagr_surf.inp	C3D20 elements using surface-to-surface contact.
eia04faf.inp	CAX4 elements.
eia08faf.inp	CAX8 elements.
eia08faf_auglagr.inp	CAX8 elements.
eia03fgf.inp	CGAX3 elements.
eia04fgf.inp	CGAX4 elements.
eia06fgf.inp	CGAX6 elements.
eia06fgf_surf.inp	CGAX6 elements using surface-to-surface contact.
eia06fgmf.inp	CGAX6M elements.
eia08fgf.inp	CGAX8 elements.
ei204fef.inp	CPE4 elements.
ei206fmf.inp	CPE6M elements.
ei206fhf.inp	CPE6MH elements.
ei208fef.inp	CPE8 elements.
ei208fef_auglagr.inp	CPE8 elements.

ei206fsf.inp	CPS6M elements.
ei312fcf.inp	CCL12, MCL6 elements.
ei312fcf_surf.inp	CCL12, MCL6 elements using surface-to-surface contact.
ei324fcf.inp	CCL24, MCL6 elements.
ei324fcf_surf.inp	CCL24, MCL6 elements using surface-to-surface contact.
ei306fcf.inp	MCL6, MCL9 elements.
ei306fcf_surf.inp	MCL6, MCL9 elements using surface-to-surface contact.
ei309fcf.inp	MCL9 elements.
ei309fcf_surf.inp	MCL9 elements using surface-to-surface contact.

Verification of the tube-tube interface element approach:

ei21stce.inp	ITT21 elements, external contact.
ei21stci.inp	ITT21 elements, internal contact.
ei31stce.inp	ITT31 elements, external contact.
ei31stci.inp	ITT31 elements, internal contact.

Finite-sliding contact between a deformable body and a rigid surface

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

B21 B22 B31 B32
PIPE21 PIPE31
C3D6 C3D8 C3D8P C3D8RP C3D10M C3D10MH C3D10MP C3D10MPH C3D27
CAX4 CAX4RP CAX6M CAX6MH CAX6MP CAX8
CGAX3 CGAX4 CGAX6 CGAX6M CGAX8
CPE4 CPE4P CPE6M CPE6MH CPE6MP CPE8 CPEG6M CPEG6MH
MCL6 MCL9

Features tested

Contact pair

An element-based surface or a node-based surface on a deformable body, and a rigid surface

Problem description

The Abaqus/Standard models consist of a solid or beam element that is resting on a rigid surface, or in the case of three-dimensional solid elements, a distance of one unit away from the rigid surface. In the latter case a displacement is applied in the first step to bring the body in contact with the rigid surface. Frictionless contact is assumed. With contact established, a downward pressure is applied on the deformable elements, resulting in contact pressures and stresses in the solid elements. Two pressure load steps are performed. The first step is a geometrically linear analysis, whereas the second step invokes geometric nonlinearity, which takes the increased contact area into account. The contact pressure (CPRESS) should balance the applied pressure load in both steps.

For axisymmetric elements with twist the test consists of five steps. Initially the solid element interferes with the rigid surface. This overclosure is removed in the first step, which is a nonlinear step. The next three steps are linear perturbation steps, wherein relative sliding and/or twisting is performed between the two contact surfaces. The following nonlinear step combines relative sliding and twisting between the two surfaces.

In the last three steps three linear perturbation analyses are conducted: a direct-solution steady-state dynamic analysis of the two bodies in contact subjected to a harmonic distributed loading, a natural frequency extraction analysis, and a subspace-based steady-state dynamic analysis.

For cylindrical membrane elements the models consist of two concentric cylinders. The deformable cylinder, which is meshed with cylindrical membrane elements, has a radius of one unit. The rigid cylinder, modeled using an analytical rigid surface, has a radius of 1.2 units. The tests consist of three steps. Initially the cylindrical membrane elements interfere with the rigid surface. This overclosure is removed in the first step, which is a linear step. The value of friction is changed in the second nonlinear step. In the final step relative sliding is performed between the two contact surfaces.

The Abaqus/Explicit model consists of a single beam element contacting an analytically rigid surface. The analysis has two steps. In the first step the contact is established, with a frictionless contact pair definition, using node-based surfaces on the deformable beam. In the next step the contact pair is redefined with friction, and the

beam is made to slide over the analytically rigid surface. Consistent contact stresses are obtained for beam and pipe elements.

Model:

Length of beams	1.0
2D solid element dimensions	5×5
3D solid element dimensions	
C3D8 (contact node set)	contact area = 5×1
C3D6	irregular: contact area = 3
C3D8	irregular: contact area = 4
C3D27	contact area = 3×2

Material:

Young's modulus	30×10^6
Poisson's ratio	0.3

Results and discussion

For tests with applied pressure, the contact pressure balances the applied downward pressure load on the deformable elements exactly.

Input files

Abaqus/Standard input files

ei22srsb.inp	B21 elements.
ei22srsb_surf.inp	B21 elements using surface-to-surface contact.
ei23srsb.inp	B22 elements.
ei23srsb_surf.inp	B22 elements using surface-to-surface contact.
eib2srsb.inp	B31 elements.
eib3srsb.inp	B32 elements.
ei33srsc.inp	C3D6 elements.
ei33srsc_surf.inp	C3D6 elements using surface-to-surface contact.
ei34srsc.inp	C3D8 elements.
ei34srsc_surf.inp	C3D8 elements using surface-to-surface contact.
eig1srsc.inp	C3D8 elements, three-dimensional node-based surface.
ei34srsp_surf.inp	C3D8P elements using surface-to-surface contact.
ei34srsp_c3d8rp.inp	C3D8RP elements.
ei39srsc_c3d10m.inp	C3D10M elements.
ei39srsc_c3d10mh.inp	C3D10MH elements.
ei39srsc_c3d10mp.inp	C3D10MP elements.
ei39srsc_c3d10mph.inp	C3D10MPH elements.
ei39srsc.inp	C3D27 elements.

eia2srsa.inp	CAX4 elements.
eia2srsa_surf.inp	CAX4 elements using surface-to-surface contact.
eia2srsa_cax4rp.inp	CAX4RP elements.
eia3srsa_cax6m.inp	CAX6M elements.
eia3srsa_cax6mh.inp	CAX6MH elements.
eia3srsa_cax6mp.inp	CAX6MP elements.
eia3srsa.inp	CAX8 elements.
eia3srsa_surf.inp	CAX8 elements using surface-to-surface contact.
eia3srsa_auglagr.inp	CAX8 elements.
eia2srsg3.inp	CGAX3 elements.
eia2srsg3_surf.inp	CGAX3 elements using surface-to-surface contact.
eia2srsg4.inp	CGAX4 elements.
eia3srsg6.inp	CGAX6 elements.
eia3srsg6_surf.inp	CGAX6 elements using surface-to-surface contact.
eia3srsgm6.inp	CGAX6M elements.
eia3srsgm6_surf.inp	CGAX6M elements using surface-to-surface contact.
eia3srsg8.inp	CGAX8 elements.
eia3srsg8_surf.inp	CGAX8 elements using surface-to-surface contact.
ei22srse.inp	CPE4 elements.
eip1srse.inp	CPE4 elements, two-dimensional node-based surface.
ei22srsp.inp	CPE4P elements.
ei23srse_cpe6m.inp	CPE6M elements.
ei23srse_cpe6mh.inp	CPE6MH elements.
ei23srsp_cpe6mp.inp	CPE6MP elements.
ei23srse.inp	CPE8 elements.
ei23srse_surf.inp	CPE8 elements using surface-to-surface contact.
ei23srse_cpeg6m.inp	CPEG6M elements.
ei23srse_cpeg6mh.inp	CPEG6MH elements.
ei36srsc_mcl6.inp	MCL6 elements.
ei36srsc_mcl6_surf.inp	MCL6 elements using surface-to-surface contact.
ei39srsc_mcl9.inp	MCL9 elements.
ei39srsc_mcl9_surf.inp	MCL9 elements using surface-to-surface contact.

Abaqus/Explicit input files

cpair_beam2d_xpl.inp	B21 elements using contact pair in Abaqus/Explicit.
cpair_beam3d_xpl.inp	B31 elements using contact pair in Abaqus/Explicit.
cpair_pipe2d_xpl.inp	PIPE21 elements using contact pair in Abaqus/Explicit.
cpair_pipe3d_xpl.inp	PIPE31 elements using contact pair in Abaqus/Explicit.

Finite-sliding contact between a deformable body and a meshed rigid surface

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Features tested

**CONTACT PAIRDSURF*, *RSURFDSURF* is a surface on the deformable body, and *RSURF* is a rigid surface meshed with either rigid elements or deformable elements declared as rigid.

Two-dimensional meshed rigid surfaces

Elements tested

B21 CPS4R R2D2 T2D2

Problem description

These tests verify that two-dimensional meshed rigid surfaces are properly generated and that the surface orientation and normal smoothing are correct. The first problem involves forming an elastic beam around a closed meshed rigid surface. This closed surface can be thought of as a pipe cross-section. The second problem is similar to the first but with user-defined normals.

The surface, which is assumed to be rigid, is meshed with 2-node rigid elements. The beam, which is 6 inches long and 0.05 inches wide, is modeled with 20 CPS4R solid elements. Its original position with respect to the first rigid surface is shown in *Figure 1*. It is assumed to be elastic with a Young's modulus of 30.0×10^6 lb/in^2 and a Poisson's ratio of 0.3. Surfaces defined on the deformable body and the rigid body are paired together to enforce contact.

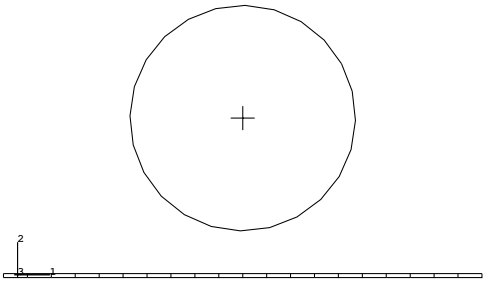


Figure 1: Original position of the beam with respect to the rigid surface.

The analysis is made up of two parts. The first part establishes contact between the beam and the rigid surface by moving the two ends of the beam upward so that contact is firmly established while constraining the beam ends horizontally. The second part involves releasing the beam end constraints and applying a pressure load to the bottom surface of the beam to mold it firmly around the pipe section. A pressure of 1000 lb/in^2 is applied in the first problem, while a pressure of 2000 lb/in^2 is applied in the second problem.

Results and discussion

The deformable body conforms to the shape of the rigid body.

Input files

ei22ssr1.inp	Two-dimensional rigid surface consisting of rigid elements with default Abaqus-generated normals.
ei22ssr1_surf.inp	Two-dimensional rigid surface consisting of rigid elements with default Abaqus-generated normals using surface-to-surface contact.
ei22ssr2.inp	Two-dimensional rigid surface consisting of rigid elements with user-specified normals.
ei22srb2.inp	Bèzier rigid surfaces used to model contact. (This capability is no longer supported.)
ei22srb2_surf.inp	Bèzier rigid surfaces used to model contact with surface-to-surface approach. (This capability is no longer supported.)
ed22ssr1.inp	Two-dimensional rigid surface consisting of beam elements declared as rigid with default Abaqus-generated normals.
ed22ssr2.inp	Two-dimensional rigid surface consisting of beam elements declared as rigid with user-specified normals.

Three-dimensional meshed rigid surfaces

Elements tested

R3D3 S3R S4 S4R

Problem description

This test verifies that three-dimensional meshed rigid surfaces are properly generated and that the search algorithm used to determine the closest distance to such surfaces is robust. The problem consists of the forming of an elastic sheet around a cylinder.

The cylinder is assumed to be rigid and has a radius of 5 inches. The original mesh with the meshed rigid surface is shown in *Figure 2*.

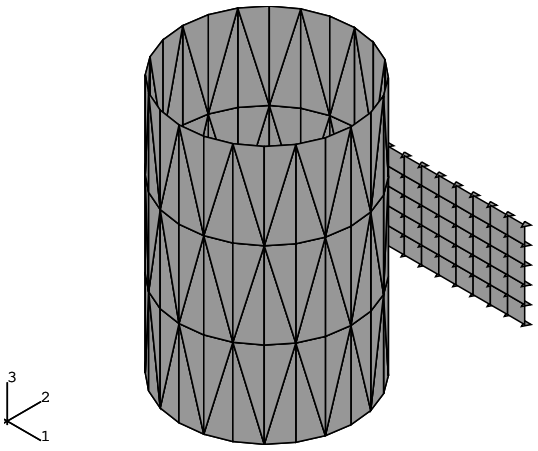


Figure 2: Original definition of cylinder.

The sheet has dimensions 10 inches by 5 inches and is modeled with fifty 4-node S4R or S4 shell elements. ENCASTRE-type boundary conditions are applied to the sheet on one side. A pressure load of 700 lb/in^2 is applied on its surface to form it around the cylinder. The sheet is assumed to be elastic with Young's modulus of 3×10^6 lb/in^2 and Poisson's ratio of 0.3. The sheet is 0.25 inches thick.

Surfaces defined on the rigid cylinder and deformable sheet are paired together to enforce contact.

Results and discussion

The deformable body conforms to the shape of the rigid body.

Input files

eig1ssr3.inp	Three-dimensional rigid surface meshed with rigid elements. Contact with S4R elements.
eig1ssr3_surf.inp	Three-dimensional rigid surface meshed with rigid elements. Surface-to-surface contact with S4R elements.
eig1ssr4.inp	Three-dimensional rigid surface meshed with rigid elements. Contact with S4 elements.
eig1srb3.inp	Bèzier rigid surfaces used to model contact. (This capability is no longer supported.)
edg1ssr3.inp	Three-dimensional rigid surface meshed with shell elements declared as rigid. Contact with S4R elements.
edg1ssr4.inp	Three-dimensional rigid surface meshed with shell elements declared as rigid. Contact with S4 elements.

Finite-sliding contact between coupled temperature-displacement elements

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D4T C3D6T C3D8IT C3D8RHT C3D8RT C3D8T C3D10MT
CAX3T CAX4RHT CAX4RT CAX4T CAX6MHT CAX6MT CAX8T CGAX4RHT
CGAX4RT CGAX4T CGAX6MHT CGAX6MT CGAX8T
CPE3T CPE4RHT CPE4RT CPE4T CPE4PT CPE4RPT CPE4PHT CPE4RPHT CPE6MHT
CPE6MT CPE8HT CPE8T CPEG4RHT CPEG4RT CPEG4T CPEG6MHT CPEG6MT
CPEG8T
CPS3T CPS4RT CPS6MT
SC8RT SC6RT S8RT SAX2T

Features tested

Contact pair

Gap conductance

Gap heat generation

Gap radiation

Problem description

The planar tests and three-dimensional tests consist of a small block pressed against a larger block that is fixed on the bottom. The smaller block slides horizontally on the larger block according to the prescribed loading and displacement history. The axisymmetric tests are essentially the same except that the sliding structures are rings; the outer ring is shorter axially than the inner ring. Relative motion in the axisymmetric tests is in the axial direction for the tests of axisymmetric elements or has axial and circumferential components for the tests of axisymmetric elements with twist. A smoothing factor of 0.05 is used on the contact pairs. For the three-dimensional tests a three-dimensional model with width 1.0 is used. The width of the bottom block is chosen slightly larger to ensure that the upper block contacts the lower block.

The mesh in *Figure 1*, used for planar tests, is representative of all meshes used in these tests.

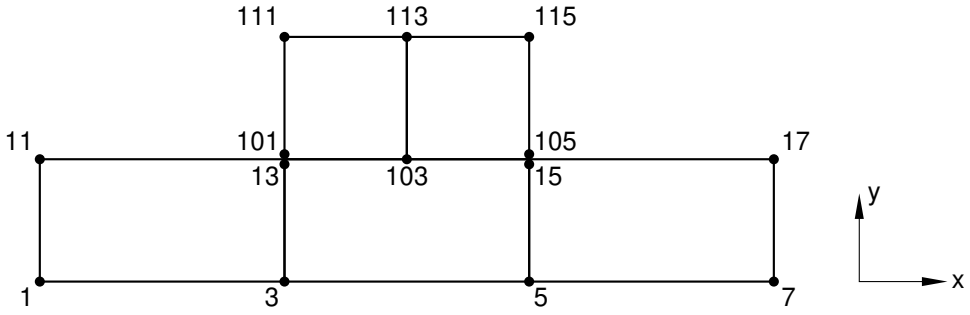


Figure 1: Representative mesh.

Material:

Solid

Linear elastic, Young's modulus = 30.0×10^6, Poisson's ratio = 0.3, conductivity = 10.0, density = 1000.0, specific heat = 0.001.

Interface

Friction coefficient (nonzero only for the frictional heat generation tests), μ=0.1.

Gap conductance varies with pressure for the interface conductance tests, $k(p$=200) = 5.0, $k(p$=100) = 20.0.

Gap conductance (for the frictional heat generation tests), 20.0.

Gap radiation constants (for the interface radiation tests only), F_A=F_B=1.0×10^{-6}, with absolute zero at θ^Z=–273.16.

Loading history for interface conductance tests (Abaqus/Standard)

Step 1, transient:

A downward pressure of 100 is applied on top of the smaller block, and a flux of 100 is applied into the smaller block through its surface. The center element of the large block has a film condition with a film coefficient of 10.0 and sink temperature of 0.0 at the bottom face. This step is used to check the gap conductivity. Results should be symmetric about an axis that is parallel to the line joining the centers of the two blocks, and thermal equilibrium must be satisfied.

The heat conducted away from the larger block via the film condition should nearly equal the heat conducted through the interface—they need not be exactly equal because transient effects are included in this step. Input file *eia2tssc.inp* illustrates the procedure to specify a time-dependent variation of the film coefficient.

Step 2, transient:

The top block is made to slide horizontally, back and forth, over the bottom block to assure that the formulation does not fail under large relative sliding. The results are consistent with thermal equilibrium. In the tests of axisymmetric elements with twist, the top block slides with circumferential motion as well.

Step 3, steady state:

The top block is in the same configuration as at the end of Step 1 but is brought to steady state to eliminate transient effects. This allows for a more exact check on thermal equilibrium of the assembly because the heat conducted across the interface must equilibrate the heat passed into the assembly by the applied flux.

Step 4, steady state:

The pressure is increased on the top surface. This is designed to test pressure-dependent interface conductivity. The temperature change across the interface should be four times that at the end of Step 3 because the interface conductivity is reduced by one-fourth.

Step 5, transient:

The applied flux is ramped down quickly, and the small block is made to slide off the larger block. This is to test that the interface heat transfer is eliminated when a secondary node slides off the end of the corresponding main surface. The smaller block becomes insulated, and the temperature is constant throughout the block.

Loading history for interface radiation tests (Abaqus/Standard)

The loading is the same for these tests as for the interface conductance tests. These problems are designed to test radiation heat transfer in the interface. Since the radiative properties are not pressure dependent, the results for Step 4 are identical to Step 3 in these runs.

Loading history for frictional heat generation tests (Abaqus/Standard)

In this analysis the top (outer) surface of the smaller block is constrained to remain straight and nonrotating via constraint equations. In this analysis the Lagrange friction formulation is used. With this formulation all relative motion is converted into heat. The default friction algorithm uses an automatic penalty method, allowing small relative motions without dissipation. In these tests this would cause the generated heat to be underestimated by about 0.7%.

Step 1:

A downward force of 200 is applied to the top surface to establish contact (an inward force of 275 is applied for the axisymmetric tests). Virtually no heat generation occurs.

Step 2:

The top block is made to slide back and forth with friction. Assuming Coulomb friction, a total of 120 units of heat is generated. Of this generated heat 60 units are absorbed by the contacting bodies because the fraction of frictional dissipation converted to heat is specified to be 0.5. Results are consistent with thermal equilibrium. In the tests of axisymmetric elements with twist, the top block slides with both axial and circumferential components of motion. The magnitude of the relative motion and the resulting heat generation is the same as in the remaining tests.

STEP 3:

The assembly sits without thermal loading to reach steady state. Because the assembly is adiabatic, it should attain a constant temperature. Based on the amount of heat generated and the heat capacity of the material, the final temperature of the assembly should be 7.5 for the planar case and 0.68 for the axisymmetric case.

Simulation with Abaqus/Explicit

A transient simulation is performed for each step. The simulation time for those steps where Abaqus/Standard performs a steady-state analysis is chosen so that enough time is allowed for the Abaqus/Explicit solution to reach steady-state conditions. Mass scaling is used to obtain an efficient solution. The rate at which the top block is forced to slide over the bottom block is reduced to ensure a quasi-static response; the amount of relative sliding between the two blocks (and, therefore, the amount of frictional heat generation, for example) is unaffected by this change.

Results and discussion

The results agree with the analytically obtained values.

Input files

Abaqus/Standard input files

Interface conductance tests:

coupledtemplgslcont_c_c3d4t_s.inp	C3D4T elements.
coupledtemplgslcont_c_c3d6t_s.inp	C3D6T elements.
coupledtemplgslcont_c_c3d8it_s.inp	C3D8IT elements.
coupledtemplgslcont_c_c3d8it_s_surf.inp	C3D8IT elements using surface-to-surface contact.
ei34tpsc_c3d8rht.inp	C3D8RHT elements.
ei34tpsc_c3d8rht_surf.inp	C3D8RHT elements using surface-to-surface contact.
ei34tpsc_c3d8rt.inp	C3D8RT elements.
ei34tpsc_c3d8rt_surf.inp	C3D8RT elements using surface-to-surface contact.
ei34tpsc.inp	C3D8T elements.
ei34tpsc_surf.inp	C3D8T elements using surface-to-surface contact.
coupledtemplgslcont_c_cax3t_s.inp	CAX3T elements.
eia2tssc_cax4rht.inp	CAX4RHT elements.
eia2tssc_cax4rt.inp	CAX4RT elements.
eia2tssc.inp	CAX4T elements.
eia2tssc_surf.inp	CAX4T elements using surface-to-surface contact.
eia3tssc_cax6mht.inp	CAX6MHT elements.
eia3tssc_cax6mt.inp	CAX6MT elements.
eia3tssc_cax6mt_surf.inp	CAX6MT elements using surface-to-surface contact.
eia3tssc.inp	CAX8T elements.
eia3tssc_surf.inp	CAX8T elements using surface-to-surface contact.
eia3tslc.inp	CAX8T, SAX2T elements.
eig2tssc_cgax4rht.inp	CGAX4RHT elements.
eig2tssc_cgax4rht_surf.inp	CGAX4RHT elements using surface-to-surface contact.
eig2tssc_cgax4rt.inp	CGAX4RT elements.
eig2tssc.inp	CGAX4T elements.
eig2tssc_surf.inp	CGAX4T elements using surface-to-surface contact.
eig3tssc_cgax6mht.inp	CGAX6MHT elements.
eig3tssc_cgax6mt.inp	CGAX6MT elements.
eig3tssc_cgax6mt_surf.inp	CGAX6MT elements using surface-to-surface contact.
eig3tssc.inp	CGAX8T elements.
coupledtemplgslcont_c_cpe3t_s.inp	CPE3T elements.
ei22tssc_cpe4rht.inp	CPE4RHT elements.
ei22tssc_cpe4rt.inp	CPE4RT elements.
ei22tssc.inp	CPE4T elements.
ei22tssc_surf.inp	CPE4T elements using surface-to-surface contact.

ei22tssc_cpe4pt.inp	CPE4PT elements.
ei22tssc_cpe4pt_surf.inp	CPE4PT elements using surface-to-surface contact.
ei22tssc_cpe4rpt.inp	CPE4RPT elements.
ei22tssc_cpe4rpt_surf.inp	CPE4RPT elements using surface-to-surface contact.
ei22tssc_cpe4pht.inp	CPE4PHT elements.
ei22tssc_cpe4pht_surf.inp	CPE4PHT elements using surface-to-surface contact.
ei22tssc_cpe4rpht.inp	CPE4RPHT elements.
ei22tssc_cpe4rpht_surf.inp	CPE4RPHT elements using surface-to-surface contact.
ei23tssc_cpe6mht.inp	CPE6MHT elements.
ei23tssc_cpe6mt.inp	CPE6MT elements.
ei28tssc.inp	CPE8HT elements.
ei23tssc.inp	CPE8T elements.
ei23tssc_surf.inp	CPE8T elements using surface-to-surface contact.
ei23tssc_auglagr.inp	CPE8T elements.
eip2tssc_cpeg4rht.inp	CPEG4RHT elements.
eip2tssc_cpeg4rht_post.inp	**POST OUTPUT* analysis.
eip2tssc_cpeg4rt.inp	CPEG4RT elements.
eip2tssc.inp	CPEG4T elements.
ei23tssc_cpeg6mht.inp	CPEG6MHT elements.
ei23tssc_cpeg6mt.inp	CPEG6MT elements.
ei23tssc_cpeg6mt_surf.inp	CPEG6MT elements using surface-to-surface contact.
eip3tssc.inp	CPEG8T elements.
eip3tssc_surf.inp	CPEG8T elements using surface-to-surface contact.
coupledtemplgslcont_c_cps3t_s.inp	CPS3T elements.
ei22tssc_cps4rt.inp	CPS4RT elements.
ei23tssc_cps6mt.inp	CPS6MT elements.

Interface radiation tests:

coupledtemplgslcont_r_c3d4t_s.inp	C3D4T elements.
coupledtemplgslcont_r_c3d6t_s.inp	C3D6T elements.
coupledtemplgslcont_r_c3d8it_s.inp	C3D8IT elements.
coupledtemplgslcont_r_c3d8it_s_surf.inp	C3D8IT elements using surface-to-surface contact.
ei34tpsr_c3d8rht.inp	C3D8RHT elements.
ei34tpsr_c3d8rht_surf.inp	C3D8RHT elements using surface-to-surface contact.
ei34tpsr_c3d8rt.inp	C3D8RT elements using surface-to-surface contact.
ei34tpsr_c3d8rt_surf.inp	C3D8RT elements.
ei34tpsr.inp	C3D8T elements.
ei34tpsr_surf.inp	C3D8T elements using surface-to-surface contact.
coupledtemplgslcont_r_cax3t_s.inp	CAX3T elements.
eia2tssr_cax4rht.inp	CAX4RHT elements.
eia2tssr_cax4rt.inp	CAX4RT elements.
eia2tssr.inp	CAX4T elements.
eia2tssr_surf.inp	CAX4T elements using surface-to-surface contact.

eia3tssr_cax6mht.inp	CAX6MHT elements.
eia3tssr_cax6mt.inp	CAX6MT elements.
eia3tssr_cax6mt_surf.inp	CAX6MT elements using surface-to-surface contact.
eia3tssr.inp	CAX8T elements.
eia3tslr.inp	CAX8T, SAX2T elements.
eig2tssr_cgax4rt.inp	CGAX4RT elements.
eig2tssr.inp	CGAX4T elements.
eig3tssr_cgax6mht.inp	CGAX6MHT elements.
eig3tssr_cgax6mt.inp	CGAX6MT elements.
eig3tssr.inp	CGAX8T elements.
coupledtemplgslcont_r_cpe3t_s.inp	CPE3T elements.
ei22tssr_cpe4rht.inp	CPE4RHT elements.
ei22tssr_cpe4rt.inp	CPE4RT elements.
ei22tssr.inp	CPE4T elements.
ei22tssr_surf.inp	CPE4T elements using surface-to-surface contact.
ei22tssr_cpe4pt.inp	CPE4PT elements.
ei22tssr_cpe4pt_surf.inp	CPE4PT elements using surface-to-surface contact.
ei22tssr_cpe4rpt.inp	CPE4RPT elements.
ei22tssr_cpe4rpt_surf.inp	CPE4RPT elements using surface-to-surface contact.
ei22tssr_cpe4pht.inp	CPE4PHT elements.
ei22tssr_cpe4pht_surf.inp	CPE4PHT elements using surface-to-surface contact.
ei22tssr_cpe4rpht.inp	CPE4RPHT elements.
ei22tssr_cpe4rpht_surf.inp	CPE4RPHT elements using surface-to-surface contact.
ei23tssr_cpe6mht.inp	CPE6MHT elements.
ei23tssr_cpe6mt.inp	CPE6MT elements.
ei23tssr.inp	CPE8T elements.
ei23tssr_surf.inp	CPE8T elements using surface-to-surface contact.
ei23tssr_auglagr.inp	CPE8T elements.
eip2tssr_cpeg4rht.inp	CPEG4RHT elements.
eip2tssr_cpeg4rt.inp	CPEG4RT elements.
eip2tssr.inp	CPEG4T elements.
ei23tssr_cpeg6mht.inp	CPEG6MHT elements.
ei23tssr_cpeg6mt.inp	CPEG6MT elements.
eip3tssr.inp	CPEG8T elements.
eip3tssr_surf.inp	CPEG8T elements using surface-to-surface contact.
coupledtemplgslcont_r_cps3t_s.inp	CPS3T elements.
ei22tssr_cps4rt.inp	CPS4RT elements.
ei23tssr_cps6mt.inp	CPS6MT elements.
ei23tssr_cps6mt_surf.inp	CPS6MT elements using surface-to-surface contact.
ei38tssr.inp	S8RT elements.
ei38tssr_surf.inp	S8RT elements using surface-to-surface contact.

Frictional heat generation tests:

coupledtemplgslcont_f_c3d4t_s.inp	C3D4T elements.
coupledtemplgslcont_f_c3d6t_s.inp	C3D6T elements.
coupledtemplgslcont_f_c3d8it_s.inp	C3D8IT elements.
coupledtemplgslcont_f_c3d8it_s_surf.inp	C3D8IT elements using surface-to-surface contact.
ei34tpsf_c3d8rht.inp	C3D8RHT elements.
ei34tpsf_c3d8rht_surf.inp	C3D8RHT elements using surface-to-surface contact.
ei34tpsf_c3d8rt.inp	C3D8RT elements.
ei34tpsf.inp	C3D8T elements.
ei34tpsf_surf.inp	C3D8T elements using surface-to-surface contact.
coupledtemplgslcont_f_cax3t_s.inp	CAX3T elements.
eia2tssf_cax4rht.inp	CAX4RHT elements.
eia2tssf_cax4rt.inp	CAX4RT elements.
eia2tssf.inp	CAX4T elements.
eia3tssf_cax6mht.inp	CAX6MHT elements.
eia3tssf_cax6mt.inp	CAX6MT elements.
eia3tssf.inp	CAX8T elements.
eia3tslf.inp	CAX8T, SAX2T elements.
eig2tssf.inp	CGAX4T elements.
eig2tssf_surf.inp	CGAX4T elements using surface-to-surface contact.
eig3tssf_cgax6mt.inp	CGAX6MT elements.
eig3tssf.inp	CGAX8T elements.
coupledtemplgslcont_f_cpe3t_s.inp	CPE3T elements.
ei22tssf_cpe4rht.inp	CPE4RHT elements.
ei22tssf_cpe4rt.inp	CPE4RT elements.
ei22tssf.inp	CPE4T elements.
ei22tssf_surf.inp	CPE4T elements using surface-to-surface contact.
ei22tssf_cpe4pt.inp	CPE4PT elements.
ei22tssf_cpe4pt_surf.inp	CPE4PT elements using surface-to-surface contact.
ei22tssf_cpe4rpt.inp	CPE4RPT elements.
ei22tssf_cpe4rpt_surf.inp	CPE4RPT elements using surface-to-surface contact.
ei22tssf_cpe4pht.inp	CPE4PHT elements.
ei22tssf_cpe4pht_surf.inp	CPE4PHT elements using surface-to-surface contact.
ei22tssf_cpe4rpht.inp	CPE4RPHT elements.
ei22tssf_cpe4rpht_surf.inp	CPE4RPHT elements using surface-to-surface contact.
ei23tssf_cpe6mht.inp	CPE6MHT elements.
ei23tssf_cpe6mt.inp	CPE6MT elements.
ei23tssf.inp	CPE8T elements.
ei23tssf_auglagr.inp	CPE8T elements.
eip2tssf.inp	CPEG4T elements.
eip3tssf.inp	CPEG8T elements.
coupledtemplgslcont_f_cps3t_s.inp	CPS3T elements.
ei22tssf_cps4rt.inp	CPS4RT elements.
ei23tssf_cps6mt.inp	CPS6MT elements.

Abaqus/Explicit input files

Interface conductance tests, mechanical contact:

coupledtemplgslcont_c_cax3t.inp	CAX3T elements.
coupledtemplgslcont_c_cax4rt.inp	CAX4RT elements.
coupledtemplgslcont_c_cax6mt.inp	CAX6MT elements.
coupledtemplgslcont_c_cpe3t.inp	CPE3T elements.
coupledtemplgslcont_c_cpe4rt.inp	CPE4RT elements.
coupledtemplgslcont_c_cpe6mt.inp	CPE6MT elements.
coupledtemplgslcont_c_cps3t.inp	CPS3T elements.
coupledtemplgslcont_c_cps4rt.inp	CPS4RT elements.
coupledtemplgslcont_c_cps6mt.inp	CPS6MT elements.
coupledtemplgslcont_c_c3d4t.inp	C3D4T elements.
coupledtemplgslcont_c_c3d6t.inp	C3D6T elements.
coupledtemplgslcont_c_c3d8rt.inp	C3D8RT elements.
coupledtemplgslcont_c_c3d8t.inp	C3D8T elements.
coupledtemplgslcont_c_c3d10mt.inp	C3D10MT elements.
coupledtemplgslcont_c_sc8rt.inp	SC8RT elements.

Interface radiation tests, mechanical contact:

coupledtemplgslcont_r_cax3t.inp	CAX3T elements.
coupledtemplgslcont_r_cax4rt.inp	CAX4RT elements.
coupledtemplgslcont_r_cax6mt.inp	CAX6MT elements.
coupledtemplgslcont_r_cpe3t.inp	CPE3T elements.
coupledtemplgslcont_r_cpe4rt.inp	CPE4RT elements.
coupledtemplgslcont_r_cpe6mt.inp	CPE6MT elements.
coupledtemplgslcont_r_cps3t.inp	CPS3T elements.
coupledtemplgslcont_r_cps4rt.inp	CPS4RT elements.
coupledtemplgslcont_r_cps6mt.inp	CPS6MT elements.
coupledtemplgslcont_r_c3d4t.inp	C3D4T elements.
coupledtemplgslcont_r_c3d6t.inp	C3D6T elements.
coupledtemplgslcont_r_c3d8rt.inp	C3D8RT elements.
coupledtemplgslcont_r_c3d8t.inp	C3D8T elements.
coupledtemplgslcont_r_c3d10mt.inp	C3D10MT elements.
coupledtemplgslcont_r_sc8rt.inp	SC8RT elements.

Frictional heat generation tests, mechanical contact:

coupledtemplgslcont_f_cax3t.inp	CAX3T elements.
coupledtemplgslcont_f_cax4rt.inp	CAX4RT elements.
coupledtemplgslcont_f_cax6mt.inp	CAX6MT elements.
coupledtemplgslcont_f_cpe3t.inp	CPE3T elements.
coupledtemplgslcont_f_cpe4rt.inp	CPE4RT elements.
coupledtemplgslcont_f_cpe6mt.inp	CPE6MT elements.
coupledtemplgslcont_f_cps3t.inp	CPS3T elements.
coupledtemplgslcont_f_cps4rt.inp	CPS4RT elements.

coupledtemplgslcont_f_cps6mt.inp	CPS6MT elements.
coupledtemplgslcont_f_c3d4t.inp	C3D4T elements.
coupledtemplgslcont_f_c3d6t.inp	C3D6T elements.
coupledtemplgslcont_f_c3d8rt.inp	C3D8RT elements.
coupledtemplgslcont_f_c3d8t.inp	C3D8T elements.
coupledtemplgslcont_f_c3d10mt.inp	C3D10MT elements.
coupledtemplgslcont_f_sc8rt.inp	SC8RT elements.

Finite-sliding contact between coupled thermal-electrical-structural elements

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

Q3D4 Q3D6 Q3D8 Q3D8R QAX4 QAX6M QAX8

Features tested

Contact pair

Gap conductance

Gap electrical conductance

Gap heat generation

Gap radiation

Problem description

The tests consist of a small block pressed against a larger block that is fixed on the bottom. The smaller block slides horizontally on the larger block according to the prescribed loading and displacement history. The axisymmetric tests are essentially the same except that the sliding structures are rings; the outer ring is shorter axially than the inner ring. Relative motion in the axisymmetric tests is in the axial direction for the tests of axisymmetric elements. A smoothing factor of 0.05 is used on the contact pairs. A three-dimensional model with width 1.0 is used. The width of the bottom block is chosen to be slightly larger than that of the upper block to ensure that the upper block contacts the lower block.

The mesh in *Figure 1*, used for axisymmetric tests, is representative of all meshes used in these tests.

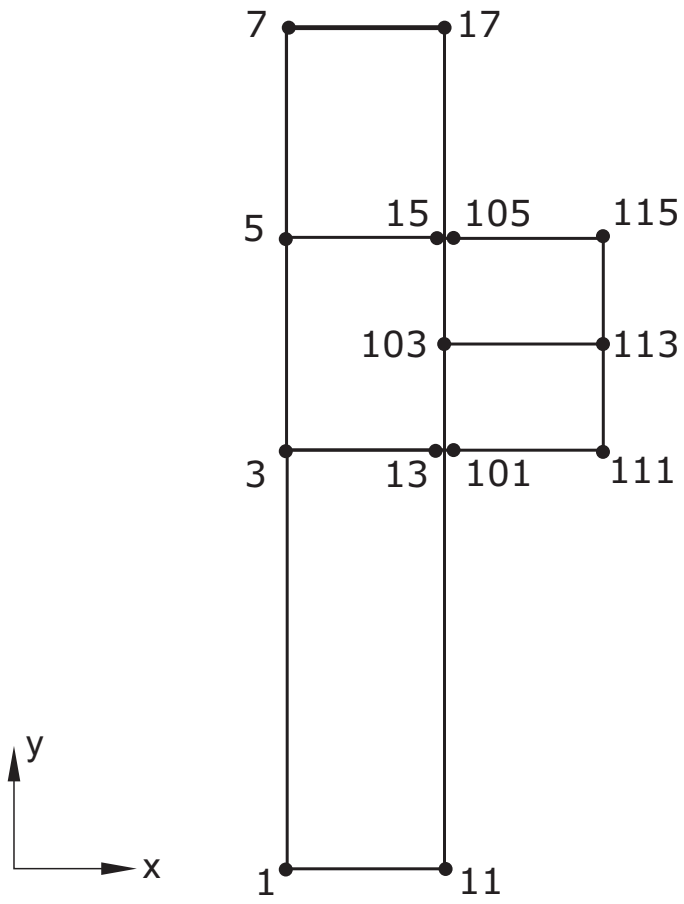


Figure 1: Representative mesh.

Material:

Solid

Linear elastic, Young's modulus = 30.0×10^6, Poisson's ratio = 0.3, conductivity = 10.0, electrical conductivity = 0.1, Joule heat fraction = 0.0, density = 1000.0, specific heat = 0.001.

Interface

Friction coefficient (nonzero only for the frictional heat generation tests), $\mu = 0.1$.

Gap conductance varies with pressure for the interface conductance tests, $k(p = 200) = 5.0$, $k(p = 100) = 20.0$.

Gap electrical conductance varies with pressure for the interface conductance tests, $k(p = 200) = 0.05$, $k(p = 100) = 0.2$.

Gap conductance (for the frictional heat generation tests), 20.0.

Gap radiation constants (for the interface radiation tests only), $\epsilon_A = \epsilon_B = 0.74074$, $F = 1.0$ with absolute zero at $\theta^Z = -273.16$.

Loading history for interface conductance tests

Step 1, transient:

A downward pressure of 100 is applied on top of the smaller block. A flux of 100 and a current flux of 1.0 are applied into the smaller block through its surface. The center element of the large block has a film condition with a film coefficient of 10.0 and sink temperature of 0.0 at the bottom face. This step is used to check the gap conductivity and the gap electrical conductivity. Results should be symmetric about an axis that is parallel to the line joining the centers of the two blocks, and thermal and electrical equilibrium must be satisfied.

Step 2, transient:

The top block is made to slide horizontally, back and forth, over the bottom block to assure that the formulation does not fail under large relative sliding. The results are consistent with thermal and electrical equilibrium.

Step 3, steady state:

The top block is in the same configuration as at the end of Step 1 but is brought to steady state to eliminate transient effects. This step allows for a more exact check on thermal equilibrium of the assembly because the heat conducted across the interface must equilibrate the heat passed into the assembly by the applied flux.

Step 4, steady state:

The pressure is increased on the top surface. This step is designed to test pressure-dependent interface conductivities. The temperature and the electrical potential changes across the interface should be four times that at the end of Step 3 because the interface conductivities are reduced by one-fourth.

Step 5, transient:

The applied flux is ramped down quickly, and the small block is made to slide off the larger block. This step tests that the interface heat transfer and current flow are eliminated when a secondary node slides off the end of the corresponding main surface. The smaller block becomes insulated, and the temperature and the electrical potential are constant throughout the block.

Loading history for interface radiation tests

The loading is the same for these tests as for the interface conductance tests, except for the value of the electrical potential, which is now set to zero at all nodes. These problems are designed to test radiation heat transfer in the interface. Since the radiative properties are not pressure dependent, the results for Step 4 are identical to those in Step 3 in these runs.

Loading history for frictional heat generation tests

The value of the electrical potential is set to zero at all nodes, and the top (outer) surface of the smaller block is constrained to remain straight and nonrotating via constraint equations. The Lagrange friction formulation is used. With this formulation all relative motion is converted into heat. The default friction algorithm uses an automatic penalty method, allowing small relative motions without dissipation. Using the default friction algorithm would cause the generated heat to be underestimated by about 0.7%.

Step 1:

A downward force of 200 is applied to the top surface to establish contact (an inward force of 275 is applied for the axisymmetric tests). Virtually no heat generation occurs.

Step 2:

The top block is made to slide back and forth with friction. Assuming Coulomb friction, a total of 120 units of heat is generated. Of this generated heat, 60 units are absorbed by the contacting bodies because the fraction of frictional dissipation converted to heat is specified to be 0.5. Results are consistent with thermal equilibrium.

STEP 3:

The assembly sits without thermal loading to reach steady state. Because the assembly is adiabatic, it should attain a constant temperature. Based on the amount of heat generated and the heat capacity of the material, the final temperature of the assembly should be 7.5 for the planar case and 0.68 for the axisymmetric case.

Results and discussion

The results agree with the analytically obtained values.

Input files

Abaqus/Standard input files

Interface conductance tests:

tes_lgslcond_q3d4.inp	Q3D4 elements.
tes_lgslcond_q3d6.inp	Q3D6 elements.
tes_lgslcond_q3d8.inp	Q3D8 elements.
tes_lgslcond_q3d8_surf.inp	Q3D8 elements using surface-to-surface contact.
tes_lgslcond_q3d8r.inp	Q3D8R elements.
tes_lgslcond_q3d8r_surf.inp	Q3D8R elements using surface-to-surface contact.
tes_lgslcont_qax4.inp	QAX4 elements.
tes_lgslcont_qax4_surf.inp	QAX4 elements using surface-to-surface contact.
tes_lgslcont_qax6m.inp	QAX6M elements.
tes_lgslcont_qax8.inp	QAX8 elements.

Interface radiation tests:

tes_lgslrad_q3d4.inp	Q3D4 elements.
tes_lgslrad_q3d6.inp	Q3D6 elements.
tes_lgslrad_q3d8.inp	Q3D8 elements.
tes_lgslrad_q3d8_surf.inp	Q3D8 elements using surface-to-surface contact.
tes_lgslrad_q3d8r.inp	Q3D8R elements.
tes_lgslrad_q3d8r_surf.inp	Q3D8R elements using surface-to-surface contact.
tes_lgslrad_qax4.inp	QAX4 elements.
tes_lgslrad_qax4_surf.inp	QAX4 elements using surface-to-surface contact.

tes_lgslrad_qax6m.inp	QAX6M elements.
tes_lgslrad_qax8.inp	QAX8 elements.

Frictional heat generation tests:

tes_lgslfrichg_q3d4.inp	Q3D4 elements.
tes_lgslfrichg_q3d6.inp	Q3D6 elements.
tes_lgslfrichg_q3d8.inp	Q3D8 elements.
tes_lgslfrichg_q3d8_surf.inp	Q3D8 elements using surface-to-surface contact.
tes_lgslfrichg_q3d8r.inp	Q3D8R elements.
tes_lgslfrichg_q3d8r_surf.inp	Q3D8R elements using surface-to-surface contact.
tes_lgslfrichg_qax4.inp	QAX4 elements.
tes_lgslfrichg_qax6m.inp	QAX6M elements.
tes_lgslfrichg_qax8.inp	QAX8 elements.

Finite-sliding contact between coupled pore pressure-displacement elements

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

CPE4P CPE4PT CPE4RPT CPE4PHT CPE4RPHT CPE6MP CPE8P
C3D8P C3D8RP C3D8PT C3D8RPT C3D20P
C3D4PH C3D10P C3D10PH C3D10MP C3D10PT C3D10PHT C3D10MPT
CAX4P CAX4RP CAX4PT CAX6MP

Features tested

Contact pair

Problem description

Two series of tests each consisting of five input files are documented. In the first series a small block is pressed against a larger block that is fixed on the bottom. The smaller block slides horizontally on the larger block according to the prescribed loading and displacement history to test the formulation in large relative sliding. The axisymmetric tests are essentially the same except that the sliding structures are rings; the outer ring is shorter axially than the inner ring, and the sliding is in the axial direction. The mesh shown in *Figure 1*, which is used to test element CPE4P, is representative of all meshes used in these tests.

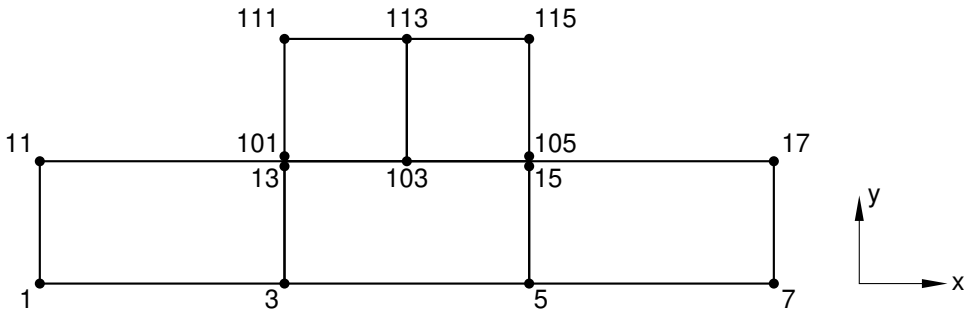


Figure 1: Representative mesh for the sliding tests.

In the second series of tests two identical blocks are pressed against each other while no sliding occurs. Fixed boundary conditions for the pore pressure degrees of freedom on the edges away from the contact interface enable the exact calculation of the pore pressure on the contact interface. The mesh shown in *Figure 2*, which is used to test element CPE4P, is representative of all meshes used in these tests.

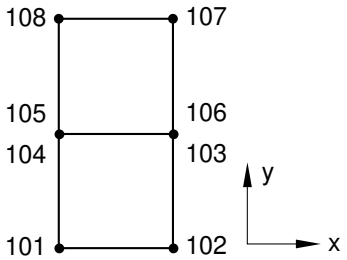


Figure 2: Representative mesh for the nonsliding tests.

Material: Linear elastic, Young's modulus = 30.0×10^6, Poisson's ratio = 0.0, permeability = 1.0×10^{-4}.

Loading history for sliding tests

Step 1, transient:

A downward pressure of 100 is applied on top of the smaller block. For the two- and three-dimensional tests a pore fluid volume flux of 3×10^{-4} is applied into the smaller block through its upper surface (area is two units). To create a constant flux through the contact interface, a pore fluid volume flux of 1×10^{-4} is applied out of the larger block lower surface (area is six units). Results should be symmetric about an axis that is parallel to the line joining the centers of the two blocks, and the total pore fluid volume flux through the contact interface should be 6×10^{-4}.

For the axisymmetric tests a pore fluid volume flux of 1×10^{-4} is applied into the smaller block through its outer surface (area is 12π), and a pore fluid volume flux of 1×10^{-4} is applied out of the larger block inner surface (area is 12π). The total pore fluid volume flux through the contact interface should be 3.76×10^{-3}.

Step 2, transient:

The top block is made to slide horizontally (1.5 units) over the bottom block. The total pore fluid volume flux through the contact interface should remain 6×10^{-4} in the two- and three-dimensional tests and 3.76×10^{-3} in the axisymmetric cases.

Loading history for the nonsliding tests

A downward pressure of 10.0 is applied on top of the upper block. The pore pressure is fixed and equal to 2.0 on the top surface of the upper block. The pore pressure on the bottom surface of the lower block is fixed and equal to 1.0. A coupled pore pressure analysis is conducted, and the pressure on the contact interface should be 1.5 for the two- and three-dimensional tests and 1.375 for the axisymmetric case.

Results and discussion

The results agree with the analytically obtained values.

Input files

Sliding tests:

ei22pfss.inp	CPE4P elements.
ei22pfss_surf.inp	CPE4P elements using surface-to-surface contact.

ei22pfss_cpe4pt.inp	CPE4PT elements.
ei22pfss_cpe4pt_surf.inp	CPE4PT elements using surface-to-surface contact.
ei22pfss_cpe4rpt.inp	CPE4RPT elements.
ei22pfss_cpe4rpt_surf.inp	CPE4RPT elements using surface-to-surface contact.
ei22pfss_cpe4pht.inp	CPE4PHT elements.
ei22pfss_cpe4pht_surf.inp	CPE4PHT elements using surface-to-surface contact.
ei22pfss_cpe4rpht.inp	CPE4RPHT elements.
ei22pfss_cpe4rpht_surf.inp	CPE4RPHT elements using surface-to-surface contact.
ei23pfss_cpe6mp.inp	CPE6MP elements.
ei23pfss_cpe6mp_surf.inp	CPE6MP elements using surface-to-surface contact.
ei23pfss.inp	CPE8P elements.
ei23pfss_auglagr.inp	CPE8P elements.
ei34pfss.inp	C3D8P elements.
ei34pfss_surf.inp	C3D8P elements using surface-to-surface contact.
ei34pfss_c3d8rp.inp	C3D8RP elements.
ei34ptfss.inp	C3D8PT elements.
ei34ptfss_surf.inp	C3D8PT elements using surface-to-surface contact.
ei34pfss_c3d8rpt.inp	C3D8RPT elements.
ei39ptfss_surf_c3d10p.inp	C3D10P elements using surface-to-surface contact.
ei39ptfss_surf_c3d10ph.inp	C3D10PH elements using surface-to-surface contact.
ei39pfss.inp	C3D10MP elements.
ei39pfss_surf.inp	C3D10MP elements using surface-to-surface contact.
ei39ptfss_surf_c3d10pt.inp	C3D10PT elements using surface-to-surface contact.
ei39ptfss_surf_c3d10pht.inp	C3D10PHT elements using surface-to-surface contact.
ei39ptfss.inp	C3D10MPT elements.
ei39ptfss_surf.inp	C3D10MPT elements using surface-to-surface contact.
ei38pfss.inp	C3D20P elements.
ei38pfss_auglagr.inp	C3D20P elements.
eia2pfss.inp	CAX4P elements.
eia2pfss_surf.inp	CAX4P elements using surface-to-surface contact.
eia2prss.inp	CAX4RP elements.
eia2ptfss.inp	CAX4PT elements.
eia2ptfss_surf.inp	CAX4PT elements using surface-to-surface contact.
eia3pfss_cax6mp.inp	CAX6MP elements.

Nonsliding tests:

ei22pfsn.inp	CPE4P elements.
ei22pfsn_cpe4pt.inp	CPE4PT elements.
ei22pfsn_cpe4rpt.inp	CPE4RPT elements.
ei22pfsn_cpe4pht.inp	CPE4PHT elements.
ei22pfsn_cpe4rpht.inp	CPE4RPHT elements.
ei23pfsn_cpe6mp.inp	CPE6MP elements.
ei23pfsn.inp	CPE8P elements.

ei23pfsn_auglagr.inp	CPE8P elements.
ei34pfsn.inp	C3D8P elements.
ei34ptfsn.inp	C3D8PT elements.
c3d4ph_ei39ptfsn.inp	C3D10PH elements.
ei39pfsn.inp	C3D10MP elements.
ei39ptfsn.inp	C3D10MPT elements.
ei38pfsn.inp	C3D20P elements.
ei38pfsn_auglagr.inp	C3D20P elements.
eia2pfsn.inp	CAX4P elements.
eia2prsn.inp	CAX4RP elements.
eia2ptfsn.inp	CAX4PT elements.

Finite sliding between concentric cylinders—axisymmetric and CAXA models

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

ISL21A

Features tested

Contact modeling in asymmetric-axisymmetric problems

Contact pair

Slide line elements

Surface on a deformable body, and surface on a deformable body or a rigid surface

Problem description

This example illustrates the use of Abaqus slide line elements and contact surface definitions in an axisymmetric structure that may undergo nonlinear, nonaxisymmetric deformation. This contact problem involves the relative motion of two outer cylinders with respect to one another and with respect to an inner, constrained cylinder. The axisymmetric model is shown in *Figure 1*, where the three cylinders are identified: the inner cylinder defined by the points ***ABHG***, the middle cylinder defined by points ***CDLK***, and the outer cylinder defined by points ***EFJI***. Two slide lines are used in this model: one along the outer edge of the inner cylinder, from node *H* through node *O*, and a second along the outer edge of the middle cylinder, from node *L* through node *D*. Axisymmetric contact elements for finite sliding (slide line elements) defined along edge ***KC*** of the middle cylinder are associated with the first slide line. Axisymmetric slide line elements defined along edge ***IE*** of the outer cylinder are associated with the second slide line.

The structure is subjected to localized pressurization to initiate contact between the surfaces in the three bodies, and then the two outer cylinders are forced to slide down the cylinder. These loading conditions are defined in two separate steps (pressurization followed by sliding). An additional perturbation step is created to test the load case definition.

In the axisymmetric model the inner cylinder is restrained from motion in the z-direction along lines ***AB*** and ***GH***. In addition, node *B* is restrained from radial motion. In the first step a pressure of 207 MPa (30×10^3 lb/in^2) is applied to edge ***JF*** of the outer cylinder, while nodes *L* and *J* are restrained vertically. During the second step the pressure is maintained, and node *L* is displaced in the negative z-direction by 127 mm (5.0 in), while node *J* is displaced in the same direction by 114.3 mm (4.5 in).

In the CGAX4 model the same steps and boundary conditions that were applied in the CAX4 model are used. An additional third step is added in which the outermost cylinder is twisted by 0.1 radians about the z-axis while the innermost cylinder is prevented from twisting.

The nonaxisymmetric model is made up of CAXA elements and additional slide line elements at various locations in the θ-direction. The area of integration for the slide line elements and the angular position (measured in degrees) of the slide line elements are defined.

In the CAXA model the boundary conditions that were applied in the axisymmetric model are kept and are extended in the θ-direction. The loading conditions are the same as the axisymmetric model. Any axisymmetric or nonaxisymmetric loading can be applied to the CAXA model after the second step.

Material:

Solid:

Young's modulus	207 GPa (30×10^6 lb/in^2)
Poisson's ratio	0.3

Coefficients of friction:

Inside edge of middle cylinder	0.2
Outer cylinder	0.6

Results and discussion

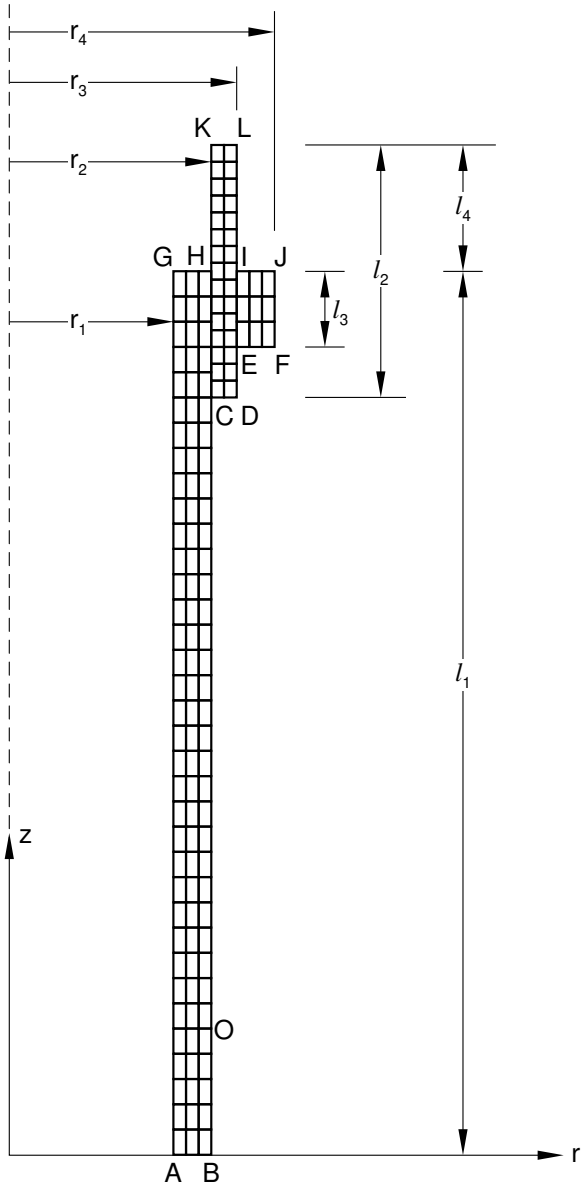
The results from the axisymmetric and three-dimensional models match.

In Step 3 of the CGAX4 analysis (in which the outermost cylinder is twisted by 0.1 radians) the middle cylinder rotates with the outermost cylinder without slipping. Relative slip is prevented by the friction that develops between the secondary/main contact pair `DSURF`/`CSURF`. However, the middle cylinder does slip with respect to the innermost cylinder. At the end of Step 2 the computed maximum torque that can be transmitted by the secondary/main surface contact pair `BSURF`/`ASURF` about the z-axis is computed to be 0.2*CTRQ=41000 lb-in, where 0.2 is the friction coefficient for the contact pair. The actual moment due to frictional shear stresses transmitted by the secondary/main surface contact pair `BSURF`/`ASURF` about the z-axis in Step 3 is 40600 lb-in, which is within 1% of the predicted value from Step 2.

Input files

eia2sssa.inp	Axisymmetric model with CAX4 elements using the contact surface approach, perturbation step with **LOAD CASE*.
eia2sssg.inp	Axisymmetric model with CGAX4 elements using the contact surface approach.
eia2ssca.inp	Axisymmetric model with ISL21A and CAX4 elements, perturbation step with **LOAD CASE*.
eia2sscn.inp	Nonaxisymmetric model with ISL21A and CAXA41 elements, perturbation step with **LOAD CASE*.

Figures



Geometry:

r_1 = 0.03302 m (1.3 in)
r_2 = 0.04064 m (1.6 in)
r_3 = 0.04572 m (1.8 in)
r_4 = 0.05334 m (2.1 in)
l_1 = 0.17780 m (7.0 in)
l_2 = 0.05080 m (2.0 in)
l_3 = 0.01524 m (0.6 in)
l_4 = 0.02540 m (1.0 in)

Slide line # 1:
defined along H–O

Slide line #2:
defined along L–D

Figure 1: Cylinder sliding model (schematic).

Self-contact of finite-sliding deformable surfaces

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

CPE3T CPE4H CPE4RT CPE6MH CPE8H CPE8HT
C3D4H C3D8H C3D10H C3D10HS C3D10MH C3D20H

Features tested

Contact pair

Surface on a deformable body that may contact itself

Problem description

The tests exercise the self-contact capability that is available for finite-sliding surfaces by declaring a single surface name in conjunction with a contact pair.

The models consist of a deformable ring with an inside radius of 2.0 and an outside radius of 3.0. The ring rests on a flat rigid surface. A circular indenter, represented by another analytical rigid surface, is initially in contact with the ring at a point. This indenter has a radius of 1.0 and is diametrically opposed to the flat surface. Contact pairs define contact between the outside surface of the ring and the two rigid surfaces and between the inside surface of the ring and itself. The ring is modeled with plane strain elements: 4-node quadrilaterals, 6-node modified triangles, or 8-node quadrilaterals. In the Abaqus/Standard simulations the elements use a hybrid formulation to accommodate an incompressible neo-Hookean hyperelastic material. Although the inside surface of the ring is closed, open surfaces are tested by eliminating one element of the inside perimeter from the surface definition, as shown in *Figure 1*.

The loading consists of two steps. In the first step the indenter moves down enough to produce self-contact of the inside surface (*Figure 2*). In the second step the indenter is simultaneously translated (−10.0 in the horizontal direction) and rotated (−8.0 around its center) in such a way that it makes the ring roll along the flat rigid surface (*Figure 3*). This produces a continuously changing region of contact. Traction is provided by setting the coefficient of friction to 0.5 for the rigid surface interfaces.

One case tests coupled thermal-mechanical interfaces. The ring is divided in two halves. The top half is given an initial temperature of 100.0, and the bottom half is given an initial temperature of 0.0. Heat transfer is allowed at the interface involving the inside surface. The two steps map into a time of 100.0 units each. This is the only case that is also solved with Abaqus/Explicit.

In the Abaqus/Explicit simulations both CPE3T and CPE4RT elements are used to model the ring; four elements are used through the thickness of the ring, and 72 elements are used around its circumference. A small amount of compressibility is added to the material definition, and mass scaling is used to obtain an efficient solution. Nondefault hourglass control is also used to control element hourglassing.

Material:

Solid: C_{10} 1.0×10^3

	D_1	1.0×10^{-3}
		(Abaqus/Explicit only)
	Conductivity	5.0×10^{-4}
	Density	1.0
	Specific heat	0.1
Self-contact interface:	Friction coefficient	0.0
	Gap conductance	5.0×10^{-4}
		(coupled temperature-displacement elements)
Rigid surface interfaces:	Friction coefficient	rough

Results and discussion

Self-contact is established and evolves over large portions of the single surface. This class of problems would be difficult to analyze with portions of the inside surface defining a conventional contact pair.

The temperature results for the coupled thermal-mechanical interface tests obtained with Abaqus/Explicit agree with those obtained with Abaqus/Standard. The stresses predicted by the two analysis products differ slightly in this case since a fully incompressible material is modeled in Abaqus/Standard while a slightly compressible one is modeled in Abaqus/Explicit.

Input files

Abaqus/Standard input files

ei24sssc.inp	CPE4H elements, closed surface.
ei24sssc_surf.inp	CPE4H elements, closed surface using surface-to-surface contact.
ei26sssc.inp	CPE6MH elements, closed surface.
ei26sssc_surf.inp	CPE6MH elements, closed surface using surface-to-surface contact.
ei28sssc.inp	CPE8H elements, closed surface.
ei28sssc_surf.inp	CPE8H elements, closed surface using surface-to-surface contact.
ei28tssc.inp	CPE8HT elements, closed surface.
ei24sssu.inp	CPE4H elements, open surface.
ei26sssu.inp	CPE6MH elements, open surface.
ei28sssu.inp	CPE8H elements, open surface.
ei34sssc.inp	C3D4H elements, closed surface.
ei34sssc_surf.inp	C3D4H elements, closed surface using surface-to-surface contact.
ei38sssc.inp	C3D8H elements, closed surface.
ei38sssc_surf.inp	C3D8H elements, closed surface using surface-to-surface contact.
ei310sssc.inp	C3D10H elements, closed surface.
ei310sssc_surf.inp	C3D10H elements, closed surface using surface-to-surface contact.
ei310isssc.inp	C3D10HS elements, closed surface.
ei310isssc_surf.inp	C3D10HS elements, closed surface using surface-to-surface contact.
ei310mssssc.inp	C3D10MH elements, closed surface.
ei310mssssc_surf.inp	C3D10MH elements, closed surface using surface-to-surface contact.

ei320sssc.inp	C3D20H elements, closed surface.
ei320sssc_surf.inp	C3D20H elements, closed surface using surface-to-surface contact.

Abaqus/Explicit input files

selfcontact_xpl_cpe3t.inp	CPE3T elements, closed surface, kinematic mechanical contact.
selfcontact_xpl_cpe4rt.inp	CPE4RT elements, closed surface, kinematic mechanical contact.
selfcontact_xpl_p_cpe4rt.inp	CPE4RT elements, closed surface, penalty mechanical contact.

Figures

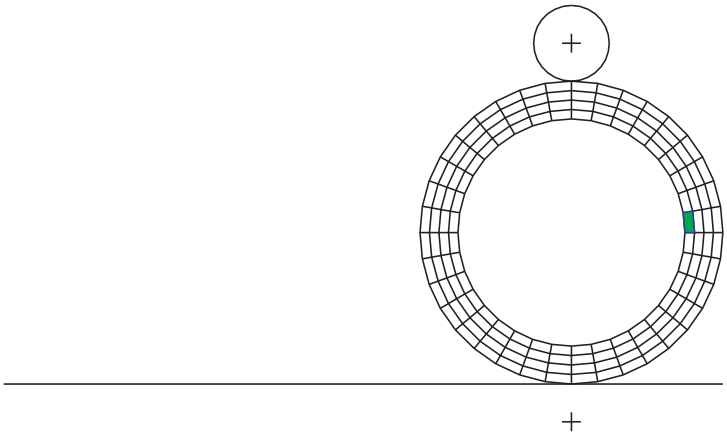


Figure 1: Self-contact model, with 4-node quads and an open surface.

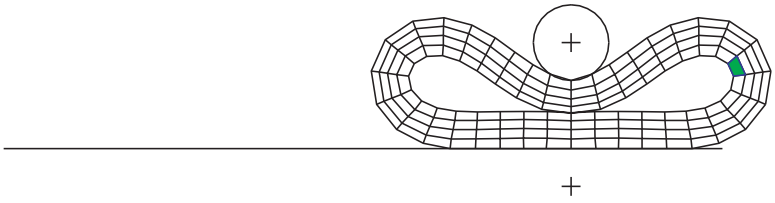


Figure 2: Deformation of Step 1.

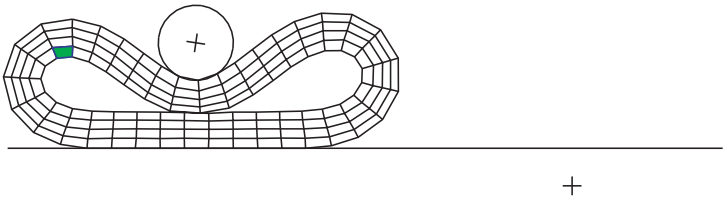


Figure 3: Deformation of Step 2.

Contact searching for analytical rigid surfaces

Products: Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

MASS

Features tested

Contact search for analytical rigid surfaces.

Problem description

A number of point masses are shot horizontally at various initial speeds and fall, due to the influence of gravity, onto a complex analytical rigid surface. The surface consists of line, circular, and parabolic segment types and includes several deep valleys to trap the point masses. The robustness of the global contact tracking algorithm is tested as Abaqus/Explicit must correctly determine throughout the analysis which main segment interacts with each secondary node. The time increment size is 0.5 s, which results in very large relative displacements for each point mass during each increment.

Results and discussion

Figure 1 shows the configuration of the point masses at various times. The contact search successfully determines the correct contact surface interactions throughout the analysis.

Input files

glb_seg_anl.inp	Two-dimensional problem.
glb_cyl_anl.inp	Three-dimensional problem.

Figures

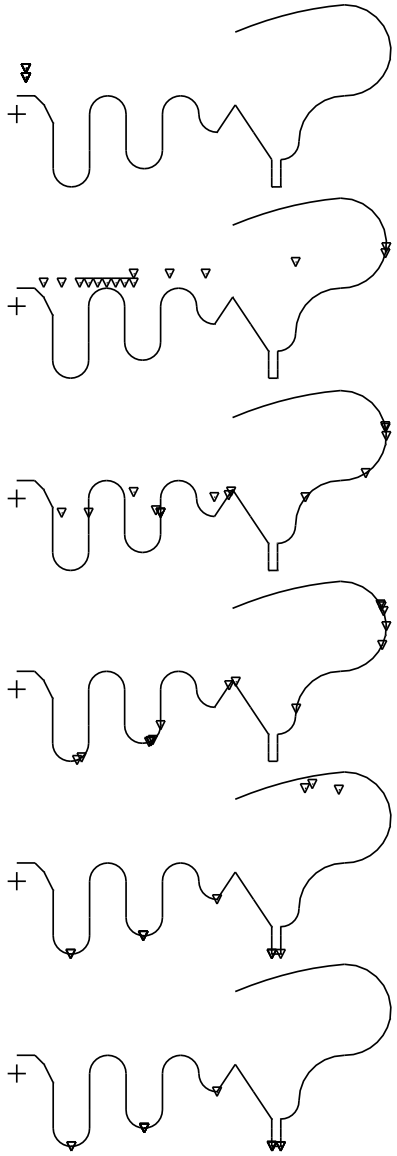


Figure 1: Configuration of the model after 0, 1, 2, 3, 11.5, and 25 seconds, respectively.

Surface-to-surface approach for finite-sliding contact

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D4 C3D8 CPE4H CPE8H GK3D6 S4R5 CGAX3T STRI65
S4R B31 C3D8R

Features tested

Surface-to-surface contact with finite sliding.

Problem description

This section deals with finite-sliding surface-to-surface contact involving stress/displacement elements. The tests utilize different surface behavior and surface interactions with the surface-to-surface approach for modeling finite-sliding contact. The tests also illustrate examples in which different facet types are involved for main and secondary surfaces.

Results and discussion

These results illustrate the accuracy and the robustness of the surface-to-surface formulation for finite-sliding contact.

Input files

Slider example:

slider_cpe4h_surf.inp	CPE4H elements modeling a slider with surface-to-surface contact.
slider_cpe8h_mpc_surf.inp	CPE8H elements modeling a slider with surface-to-surface contact.
gasket_surf.inp	GK3D6 elements contacting a slider using surface-to-surface contact.
slide-shells_surf.inp	STRI65 elements modeling a slider with surface-to-surface contact.
edg1s4r5_surf.inp	S4R5 elements modeling a slider with surface-to-surface contact.

Examples for facet type of the main surface different from the secondary:

beam-shell_surf.inp	Beam with B31 elements sliding on a shell surface with S4R elements modeled using surface-to-surface contact.
slide-shell-on-solid_surf.inp	S4R slider in contact with C3D8R element-based surface modeled using surface-to-surface contact.

Compressing and twisting one three-dimensional block on another:

block_c3d4_a_surf.inp	C3D4 elements using surface-to-surface contact.
block_c3d4_b_surf.inp	C3D4 elements using surface-to-surface contact and the unsymmetric solver to handle additional compression.

block_c3d8_a_surf.inp	C3D8 elements using surface-to-surface contact.
block_c3d8_b_surf.inp	C3D8 elements using surface-to-surface contact and the unsymmetric solver to handle additional compression.

Examples using the “path-based” tracking algorithm:

rollcyl_test.inp	Paper rolling model using double-sided shell and discrete rigid surfaces.
block_c3d8_b_surf_ntrack.inp	Compressed and twisted blocks.

General contact in Abaqus/Standard

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D4 C3D6 C3D8 C3D8R C3D10PT
CAX4R
CCL12
CPE4 CPE4H CPE8H
CPS4
DC1D2 DC1D2E DC2D4 DC2D4E DC2D8 DC2D8E
DC3D8 DC3D8R DC3D20E DCAX4 DCAX8 DCC1D2
DS3 DS4 DS6 DS8 DSAX1 DSAX2
GK3D6
MCL6
R3D4
S3 S4R

Features tested

General contact formulation in Abaqus/Standard.

Problem description

This section deals with the general contact capability in Abaqus/Standard. These examples use different type of elements and test different features used in general contact.

Results and discussion

These results illustrate the accuracy of the general contact formulation in Abaqus/Standard.

Input files

Examples for contact initialization assignment:

gcontact_shell_op1.inp	Shells with S4R and S3 elements with initial opening: default behavior.
gcontact_shell_op2.inp	Shells with S4R and S3 elements with initial opening: resolve opening.
gcontact_shell_op3.inp	Shells with S4R and S3 elements with initial opening: remain open.
gcontact_shell_op_ss4.inp	Shells with S4R and S3 elements with initial opening and single-sided surface: resolve opening.
gcontact_shell_op_ss5.inp	Shells with S4R and S3 elements with initial opening and single-sided surface: remain open.

gcontact_shell_op_ssSPOS6.inp	Shells with S4R and S3 elements with initial opening and single-sided surfaces 2 and 4 (SPOS).
gcontact_shell_op_ssSPOS7.inp	Shells with S4R and S3 elements with initial opening and single-sided surfaces 1 and 6 (SPOS).
gcontact_shell_op_ssSPOS8.inp	Shells with S4R and S3 elements with initial opening and single-sided surfaces SPOS (all surfaces).
gcontact_shell_ov9.inp	Shells with S4R and S3 elements with initial overclosure: default behavior.
gcontact_shell_ov10.inp	Shells with S4R and S3 elements with initial overclosure: retain overclosure.
gcontact_shell_ov11.inp	Shells with S4R and S3 elements with initial overclosure: resolve overclosure.
gcontact_cross-shell_def12.inp	Cross shells with S4R elements: default behavior.
gcontact_cross-shell_ss13.inp	Cross shells with S4R elements: single-sided surface.
gcontact_solid_op14.inp	C3D8R elements with initial opening: default behavior.
gcontact_solid_op15.inp	C3D8R elements with initial opening: resolve opening.
gcontact_solid_op16.inp	C3D8R elements with initial opening: remain open.
gcontact_solid_op17.inp	C3D8R elements with initial opening and single-sided surface: resolve opening.
gcontact_solid_op18.inp	C3D8R elements with initial opening and single-sided surface: remain open.
gcontact_solid_ov19.inp	C3D8R elements with initial overclosure: default behavior.
gcontact_solid_ov20.inp	C3D8R elements with initial overclosure: retain overclosure.
gcontact_solid_ov21.inp	C3D8R elements with initial overclosure: resolve overclosure.

Examples of surface smoothing technique for general contact:

surfsmooth_rings_gcont.inp	Shrink fit of two concentric rings, two-dimensional.
surfsmooth_rings_3d_gcont.inp	Shrink fit of two concentric rings, three-dimensional.
surfsmooth_spheres_gcont.inp	Shrink fit of two concentric spheres.
surfsmooth_cone_gcont.inp	Push down of an inner cone piece against an outer cone piece.

Examples of edge-to-surface contact:

slide_under_edge.inp	Sliding blocks with feature edges activated.
tennis_std_gcont.inp	Tennis ball hitting bracket modeled with beam-to-surface contact.
rigid_to_beam_coarse.inp	Rigid elements contacting large diameter beams.
rigid_to_beam_refined.inp	Refined rigid elements contacting large diameter beams.
shell_external_beam.inp	Bolt modeled with beams bearing against hole modeled with shells.
solid_external_beam.inp	Bolt modeled with beams bearing against hole modeled with solids.

Examples of contact with analytical rigid surface:

shelledge_to_cyl_analytic.inp	Contact between shell edges and analytical rigid surface.
edge_to_rev_analytic.inp	Simple edge to analytical surface contact.
full_compress_block_gcont.inp	Confined compression of rubber block.
ei310isssc_gcont.inp	Ring compressed and slid against an analytical rigid surface.
beamstandard2_gcont.inp	Cantilever beam modeled with shell elements.
eib3srsb_gcont.inp	Simple test with one element model.

ei36srsc_mcl6_gcont.inp	6-node cylindrical membrane elements model.
ei39srsc_mcl9_gcont.inp	9-node cylindrical membrane elements model.
ei36sirc_ccl9_gcont.inp	9-node cylindrical prism elements model.
ei36sirc_ccl12_gcont.inp	12-node cylindrical brick elements model.
ei39sirc_ccl18_gcont.inp	18-node cylindrical prism elements model.
ei39sirc_ccl24_gcont.inp	24-node cylindrical brick elements model.

General contact with heat transfer and coupled thermal electrical procedures:

cracktip2d_gc_discc.inp	DC2D4 elements using general contact with an embedded crack with gap conduction for heat transfer across the crack.
dc3d8r_gap_gc.inp	DC3D8R elements using general contact with gap conduction.
eia2discc_gc.inp	DCAX4 elements using general contact with gap conduction.
eia3discc_gc.inp	DCAX8 elements using general contact with gap conduction.
ei23discc_gc.inp	DC2D8 elements using general contact with gap conduction.
eia2disrs_gc.inp	Double-sided DSAX1 shell elements using general contact with gap radiation.
eia3disrs_gc.inp	Double-sided DSAX2 shell elements using general contact with gap radiation.
ei33discs_gc.inp	Double-sided DS3 shell elements using general contact with gap conduction.
ei33disrs_gc.inp	Double-sided DS3 shell elements using general contact with gap radiation.
ei34discs_gc.inp	Double-sided DS4 shell elements using general contact with gap conduction.
ei34disrs_gc.inp	Double-sided DS4 shell elements using general contact with gap radiation.
ei36discs_gc.inp	Double-sided DS6 shell elements using general contact with gap conduction.
ei36disrs_gc.inp	Double-sided DS6 shell elements using general contact with gap radiation.
ei38discs_gc.inp	Double-sided DS8 shell elements using general contact with gap conduction.
ei38disrs_gc.inp	Double-sided DS8 shell elements using general contact with gap radiation.
ei22vsjc_gc.inp	DC2D4E elements using general contact with gap heat, electrical conduction, and electrical heat generation.
ei23vsjr_gc.inp	DC2D8E elements using general contact with gap radiation, electrical conduction, and electrical heat generation.
ei34vsjc_gc.inp	DC3D8E elements using general contact with gap heat, electrical conduction, and electrical heat generation.
ei38vsjr_gc.inp	DC3D20E elements using general contact with gap radiation, electrical conduction, and electrical heat generation.
xrvd38n2_parallel_gc.inp	DC3D8 elements using general contact with gap conduction. The radiation heat transfer is modeled with cavity radiation.
uinterht_gc.inp	DC2D4 elements using general contact with user subroutine *UINTER* modeling heat transfer interaction behavior.
stob_heat.inp	Beam as main edge-to-surface contact with DC3D8 and DC1D2 element-based surfaces in thermal interactions.
stob_heatelec.inp	Beam as main edge-to-surface contact with DC3D8E and DC1D2E element-based surfaces in thermal and electrical interactions.
coil_beam-tube_solid_ht.inp	Beam as main edge-to-surface contact with DC3D8 and DC1D2 element-based surfaces in thermal interaction with gap conduction.
coil_beam-tube_solid_ht_ radiation.inp	Beam as main edge-to-surface contact with DC3D8 and DC1D2 element-based surfaces in thermal interaction with gap radiation.

shell_to_1d_ht.inp	Beam as main edge-to-surface contact with DS4 and DC1D2 element-based surfaces in thermal interaction.
solid_between_two_beams_ht.inp	Beam as main edge-to-surface contact with DC3D20 and DCC1D2 element-based surfaces in thermal interaction.
heat_exchanger_fine_diffconv_steady.inp	Beam as main edge-to-surface and surface-to-surface contact with DC3D8 and DCC1D2 element-based surfaces in thermal interaction in a heat exchanger model.

General contact with the small-sliding tracking approach

2d_crack_gcsmall.inp	CPS4 elements where opposing faces of a crack are in self-contact.
3d_crack_gcsmall.inp	C3D8R elements where opposing faces of a crack are in self-contact.
ei22siam_gcsmall.inp	CPE4 elements and analytical rigid surface. User-specified clearance normals and internally computed clearances.
ei22siao_gcsmall.inp	CPE4 elements and analytical rigid surface. User-specified clearances.
ei34sisc_lcp_gcsmall.inp	C3D8 elements with the LCP solution technique for frictionless small-sliding contact.
rigid_gcsmall_lcp.inp	C3D8 and R3D4 elements with the LCP solution technique for frictionless small-sliding contact.
ei36sirc_ccl12_gcsmall.inp	CCL12 cylindrical elements modeling an initial interference fit.
ei36sirc_mcl6_gcsmall.inp	MCL6 cylindrical membrane elements modeling an initial interference fit.
ei39ptfss_c3d10pt_gcsmall.inp	C3D10PT coupled elements modeling soils consolidation.
gcorr_std_gcsmall_a1.inp	S4R elements with various surface geometry corrections.
gcorr_std_gcsmall_two_coarse_solid_tori_e.inp	C3D8 and C3D6 elements with toroidal surface geometry corrections.
w_bolted_flange_gcsmall_complete.inp	C3D8R modeling a bolted flange.

Examples for step-dependent contact activated and deactivated:

vertex_solid_corner_gcontsd.inp	C3D8R elements with vertex-to-surface contact; initially active contact deactivated, reactivated later with shrink fit.
block_gc_stepmdlc_remove_gcontsd.inp	C3D8R and S4R elements with various general contact formulations that are initially active, deactivated, and reactivated in subsequent steps.
analytic_rigid_gcontsd.inp	C3D8RH elements with deformable-to-analytic rigid contact; contact activated after moving the analytic rigid surface, resulting in an overclosure resolved as a shrink fit.
cohesive-contact-gcontsd.inp	C3D8RH elements; initially inactive contact activated in a later step to activate cohesive contact.
w_radial_seal_complete_gcontsd.inp	CAX4R, CAX3H, and CAX4H elements with deformable-to-analytic rigid contact, activated and deactivated across several steps.
pull_in_gcontsd.inp	PIPE22 elements with edge-to-edge radial contact to model tube-in-tube insertion; contact activated in later step.

coh_cont_4elem_gcontsd.inp	S4 elements; initially active cohesive contact, deactivated and reactivated in a later step.
coupledtempDispPore_gcontsd.inp	C3D8PT elements with contact; activated in a later step.
truss2truss_unwind_gcontsd.inp	T3D2 elements with edge-to-edge radial contact; contact deactivated to unwind a bundle of twisted wires.
BlocksContact3_1smslid_gcontsd.inp	C3D8R elements with both small- and finite-sliding tracking in general contact; contact activated and deactivated for finite-sliding interactions.
ei34tpsc_c3d8rht_std_gcontsd.inp	C3D8RHT elements with contact deactivated in a coupled temperature-displacement analysis.
beam2beam_gcont_edge_gcontsd.inp	B31 elements with contact deactivated in a later step.
deepdrawbox_std_both_gcontsd.inp	S4R elements with contact activated and deactivated for deformable-to-analytic rigid contact.
assemble_example_gcontsd.inp	C3D8R elements with initially inactive contact activated progressively during assembly followed by loading.
append_beams_gcontsd.inp	CPE4R elements modeling two beam sections with contact initially inactive.
append_beams_gcontsd_restart.inp	Restart analysis of previous model with cohesive contact activated to attach the disconnected beam sections.

Miscellaneous:

block_c3d4_b_std_gcont.inp	C3D4 elements using general contact and the unsymmetric solver to handle additional compression.
block_c3d8_b_gcont.inp	C3D8 elements using general contact and the unsymmetric solver to handle additional compression.
block_Xcontact_Slide.inp	General contact with an initial stationary XFEM crack surface.
boltpipeflange_3d_gk3d18_gcont.inp	Three-dimensional analysis using general contact formulation containing a gasket modeled with gasket elements.
cantilever_Xcontact_propcrack.inp	General contact with a propagating XFEM crack surface.
gasket_surf_gcont.inp	GK3D6 elements contacting a slider using general contact
rollcyl_test_gcont.inp	Paper rolling model using general contact formulation.
slider_cpe4h_gcont.inp	CPE4H elements modeling a slider with general contact.
slider_cpe8h_mpc_gcont.inp	CPE8H elements modeling a slider with general contact.
sphere_finite_std_gcont_balanc.inp	Balanced main-secondary formulation with general contact for the spherical indentation problem.
sphere_finite_std_gcont_adj1.inp	Initial contact openings with strain-free adjustments for the spherical indentation problem.
sphere_finite_std_gcont_adj2.inp	Initial contact overclosures with strain-free adjustments for the spherical indentation problem.

Surface smoothing for surface-to-surface contact

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Features tested

Surface smoothing technique for surface-to-surface contact.

Problem description

Examples are given to verify the behavior of the surface smoothing technique, which helps to improve contact stress accuracy. The method applies to both finite-sliding and small-sliding surface-to-surface contact.

Results and discussion

These results show significant accuracy improvement for models subject to relatively small deformation compared to equivalent analyses without the surface smoothing technique.

Input files

surfsmooth_rings.inp	Shrink fit of two concentric rings, two-dimensional, finite-sliding.
surfsmooth_rings_3d.inp	Shrink fit of two concentric rings, three-dimensional, finite-sliding.
surfsmooth_rings_sslide.inp	Shrink fit of two concentric rings, two-dimensional, small-sliding.
surfsmooth_rings_sslide3d.inp	Shrink fit of two concentric rings, three-dimensional, small-sliding.
surfsmooth_spheres.inp	Shrink fit of two concentric spheres, three-dimensional, finite-sliding.
surfsmooth_cone.inp	Push down of an inner cone piece against an outer cone piece.

Initial contact stress

Products: Abaqus/Standard

References:

- **INITIAL CONDITIONS*

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D8 C3D8R C3D10 C3D20R CPE4 CPE4P CPE8RP CPS4

Problem description

Initial contact stresses are computed automatically from user-specified underlying element initial stresses. This capability is tested for both finite- and small-sliding contact, the node-to-surface and surface-to-surface contact formulations, and general contact. Representative element types (including XFEM elements with an initial crack) are incorporated in these tests. The enhanced geostatic procedure is used to obtain an equilibrium state of stress while minimizing displacements to be within a specified tolerance. These tests also verify enhanced geostatic procedures that involve contact interactions.

Results and discussion

In all cases the tests verify the initial contact stress capability. The initial contact stresses are reported correctly at the zero increment. In the enhanced geostatic procedure, the final displacements satisfy the prescribed tolerance.

Input files

inicstr_block_pen_both.inp
: Contact initial stresses are computed approximately at a contact interface between two rows of elements with uniform differing element stresses. Penalty contact enforcement is used with the surface-to-surface formulation. The enhanced geostatic procedure (invoked with the UTOL parameter) is specified for the first and only step.

inicstr_block_pen_c0_both.inp
: Same as `inicstr_block_pen_both.inp` except using penalty contact enforcement with a nonzero clearance at zero pressure.

inicstr_block_augm_both.inp
: Same as `inicstr_block_pen_c0_both.inp` except using augmented Lagrange contact enforcement.

inicstr_block_direct_both.inp
: Same as `inicstr_block_augm_both.inp` except using direct contact enforcement with Lagrange multipliers.

inicstr_block_presover_both.inp
: Same as `inicstr_block_direct_both.inp` except the exponential pressure-overclosure relation is used.

inicstr_wet_block.inp
: Initial effective contact stresses are derived from initial effective stresses in the underlying deformable elements for a block model with pore pressure degrees of freedom and linear two-dimensional elements.

inicstr_quadcdp2d.inp Initial effective contact stresses are derived from initial effective stresses in the underlying deformable elements for a block model with pore pressure degrees of freedom and quadratic plane strain elements.

inicstr_block_slide_xcont.inp Initial contact stresses on an initial stationary crack are derived from initial stresses in the underlying XFEM elements for a block model. The analysis involves a push down step followed by finite sliding between crack surfaces modeled with general contact.

inicstr_block_normal_ics_both.inp Initial contact stresses are calculated for a block model with linear three-dimensional hexahedral elements, and the enhanced geostatic procedure (invoked with the UTOL parameter) is used to minimize displacements.

inicstr_wet_blockrig.inp Initial effective contact stresses are derived from initial effective stresses in the underlying deformable elements on one side of a deformable to rigid contact with pore pressure degrees of freedom and linear three-dimensional elements.

inicstr_blocksuppl.inp Initial contact stresses are calculated for a block model meshed using quadratic tetrahedral elements with a node-to-surface contact pair with active supplementary constraints.

inicstr_fault2d_flat_dry_alone.inp Initial contact normal and shear stresses are derived in a two-dimensional fault slip with a linearly varying initial stress field in the underlying CPE4 elements. These initial contact stresses serve as a better starting point for establishing an overall equilibrium state.

inicstr_fault2d_flat_dry.inp Same as `inicstr_fault2d_flat_dry_alone.inp` except using the enhanced geostatic procedure (invoked with the UTOL parameter).

inicstr_fault2d_flat_wet_alone.inp Initial contact normal and shear effective stresses are derived in a two-dimensional fault slip with a linearly varying initial stress field in the underlying CPE4P elements. These initial contact stresses serve as a better starting point for establishing an overall geostatic equilibrium state.

inicstr_fault2d_flat_wet.inp Same as `inicstr_fault2d_flat_wet_alone.inp` except using the enhanced geostatic procedure (invoked with the UTOL parameter).

The following input files test initial element stresses being read from a prior analysis:

inicstr_ics_alone_patch_orig.inp A block of quadratic elements are pushed down against a flat rigid surface resulting in uniform element stresses.

inicstr_ics_alone_patch_init_sim.inp Initial element stresses are read in through the SIM database generated from `inicstr_ics_alone_patch_orig.inp`. Initial contact stresses derived from the element stresses bring the initial state to an equilibrium state.

inicstr_interf_orig.inp Cylindrical shrink fit analysis followed by an axial twist to establish element stresses.

inicstr_interf_initfromstep2.inp Read initial element stresses from the output database of `inicstr_interf_orig.inp`, and derive initial contact normal and shear contact stresses. The derived initial contact stresses bring the initial state to a nearly equilibrium state.

Contact wear modeling

Products: Abaqus/Standard Abaqus/Explicit

References:

- *Contact Wear*

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D10 C3D4 C3D8 C3D8R C3D10M C3D10MH C3D10T C3D20 C3D20R C3D27 CPE4 CPE4H CPE8 CPS3 CPS4R R3D4 S4R

Problem description

The contact wear capability is tested for both general contact and contact pairs, with different element types using the finite- and small-sliding tracking approaches. All the tests have wear properties specified in accordance with variations of Archard's wear equation. Some tests use a variant of Archard's model in which wear is proportional to the friction coefficient, and other tests use Archard's model without the friction coefficient directly appearing in the equation for incremental wear (although the wear coefficient might depend on the friction coefficient).

In addition, approximate modeling of wear accumulation over a large number of repetitive wear cycles is tested. Step cycling for wear cycle repetition and step cycling controls for early termination of wear cycles and automatic calculation of batch sizes to scale wear coefficients are tested.

Results and discussion

Output variable CWEAR is requested in all these tests for nodal wear distances. In simple cases, CWEAR is verified to match hand calculations of wear distances based on the Archard's wear model. Most of these test cases are complex enough that closed form solutions are not available.

For the test with a flat analytical rigid surface compressing and sliding over a block-shaped C3D8 element without friction in a sequence of seven steps, the stress state in the block remains uniform uniaxial with varying magnitude. Wear is not proportional to the friction coefficient in this test. Wear accumulates during increments in which the contact pressure and incremental slip are both nonzero. Contact pressure and slip increments are uniform across the nodes in contact, so the wear is also uniform. The accumulated wear distance is consistent with hand calculations.

For tests with a block modeled with C3D10 elements pressed into and sliding across a flat surface, wear is proportional to the friction coefficient. Frictional forces lead to spatial variation in the stress state within the block and spatial variation of contact pressure over the surfaces, such that incremental wear accumulation is greatest on specific edge locations of the block. Wear accumulation is consistent with the evolution of frictional stress and incremental sliding. Small differences in stress state and wear across these tests is due to numerical details, such as whether a penalty method or Lagrange multiplier method is used to enforce contact conditions.

For tests with two blocks twisting while pressed together with wear properties assigned to each face, wear accumulation is small near centers and corners of faces in contact because these regions experience relatively

small slip or are not in contact for most of the simulation. Contour plots of wear distance on contact faces show that most wear occurs within the ring region at intermediate distances from the center of these faces, where contact pressures and slip rates are often large during the simulation. The test case with linear tetrahedral elements is overly stiff for the bending-related mode that should occur for much of the twisting motion while the respective corners are not aligned. Therefore, this test overestimates contact stresses and wear distances near edges and underestimates contact stresses and wear distances away from edges.

For the test involving two hollow hemispherical parts in frictionless contact with relative spinning between two parts, wear accumulates on the contact surface of the outer part but not on the contact surface of the inner part because the wear model assigned to the outer part is not proportional to the friction coefficient and the wear model assigned to the inner part is proportional to the friction coefficient (and the friction coefficient is zero for this test). The contact pressure is relatively uniform over the surfaces in contact. The sliding distance is proportional to the distance from the axis of relative spinning between the contact surfaces. The wear distance is approximately proportional to the distance from the axis of relative spinning.

For the test involving compression of a gasket, wear develops on some regions of the gasket due to contact with a rigid surface and develops on other regions of the gasket due to self-contact. Wear accumulation is not proportional to the friction coefficient for this test, so the incremental wear distance at a particular location of the gasket surface is related to the contact pressure and the incremental slip at that location during that increment. Wear is assigned as a surface property, so the incremental wear distance at a given surface location does not depend on characteristics of the opposing surface.

For the test involving self-contact of a ring, "rolling" motion in the second step leads to significant sliding and wear where self-contact is active.

For the two tests involving wear calculations in a steady-state transport procedure, the initial run involves an axisymmetric analysis followed by a symmetric model generation of the 3D model, which in turn is used for steady-state transport wear calculations. The wear rate at any nodal location in a steady-state transport step involves the weighted sum of the wear rates the node experiences as it traverses a streamline. Therefore, the wear increments and the total wear at a node would not directly correlate with the reported contact stress and slip rates at that location.

Input files

block_c3d4_a_surf_wear.inp	Two blocks meshed with C3D4 elements are pressed together, then a torque is applied causing one block to twist relative to the other block. A finite-sliding, surface-to-surface contact pair is used to model contact. Wear properties are assigned to each contact surface.
block_c3d8_b_gcont_wear.inp	Two blocks meshed with C3D8 elements are pressed together, then a torque is applied causing one block to twist relative to the other block. General contact is used to model contact. Wear properties are assigned to each contact surface.
defo_disrigid_CP_ss_Shell_wear_orig.inp	Two blocks modeled with solid C3D10 elements are pressed into and slide back and forth (over two steps) across top and bottom sides, respectively, of a fully constrained, flat shell modeled with S4R elements. Friction coefficient dependence is explicitly specified in the wear model. Small-sliding, surface-to-surface contact pairs are used to model contact.
defo_disrigid_cp_ss_ntos_wear.inp	A block modeled with solid C3D10 elements is pressed into and slides across a flat rigid surface modeled with R3D4 elements in contact using the small-sliding, node-to-surface formulation. Friction coefficient dependence is explicitly specified in the wear model. Small-sliding, surface-to-surface contact pairs are used to model contact.
defo_disrigid_gc_ss_wear.inp	A block modeled with solid C3D10 elements is pressed into and slides across a flat rigid surface modeled with R3D4 elements. Friction coefficient dependence is explicitly specified in the wear model. General contact with the small-sliding tracking approach is used to model contact.

defo_disrigid_oneCP_ss_Shell_wear_orig.inp	A block modeled with solid C3D10 elements is pressed into and slides back and forth (over two steps) across a fully constrained flat surface modeled with shell S4R elements. Friction coefficient dependence is explicitly specified in the wear model. A small-sliding, surface-to-surface contact pair is used to model contact.
defo_disrigid_oneCP_ss_Shell_wear_model_import.inp	This simulation imports results from the end of the first step of the previous test, `defo_disrigid_oneCP_ss_Shell_wear_orig.inp`, and repeats the second step of the previous test.
ei310msssc_supplconstr_wear.inp	This simulation involves compressing a deformable ring modeled with C3D10MH elements between a rigid rod and a flat rigid surface such that opposing portions of the ring come into contact. In a second step the rod is moved in a direction parallel to the flat surface such as to cause a "rolling" motion of the compressed ring and sliding at the self-contact interface of the ring. Contact is modeled with a finite-sliding, node-to-surface contact pairs.
2block_wearT_ss.inp	A block modeled is pressed into and slide across a larger flat surface of an elongated block. Both deformable blocks are modeled with solid C3D10T elements. A coupled temperature-displacement step is performed. Friction coefficient dependence is explicitly specified in the wear model for both surfaces in contact.
defo_arig_wear.inp	This simulation involves wear accumulation due to frictionless contact between a single, block-shaped C3D8 element and an analytical rigid surface across seven steps. The wear model is independent of friction for this test. The prescribed motion of the analytical rigid surface remains parallel to the face of the block and moves in the normal or tangential direction in respective steps. A finite-sliding contact pair is used to model contact.
supplconstr_cpe8_exponential_wear.inp	A 2D block is pressed into and then briefly slides across a larger 2D block. Both blocks are modeled with CPE8 quadratic elements. A small-sliding, node-to-surface contact pair with supplementary constraints is used to model contact. Friction coefficient dependency is explicitly specified in the wear model for both surfaces in contact. An exponential contact pressure versus overclosure behavior is in effect for contact.
surfsmooth_spheres_wear.inp	Interference fit is resolved between two hollow hemispherical parts, then the inner part is rotated relative to the outer part via distributing couplings. A frictionless surface-to-surface contact pair models contact between the two parts. The wear model explicitly involves the friction coefficient for the inner part but not the outer part. Since the friction coefficient is zero in this example, wear is expected only on the inner part because incremental wear for the outer part is not proportional to the friction coefficient.
selfcontact_gask_wear.inp	A rigid surface compresses a gasket in this simulation involving CPE4H elements in self-contact with finite-sliding, node-to-surface contact pairs. A wear model that does not have wear proportional to the friction coefficient is assigned to the gasket surface.
slider_cpe4h_surf_wearcycle_stepctrl.inp	A 2D block is pressed into and slides back and forth repetitively across a larger stationary block. Both blocks are meshed with CPE4H elements and are in contact with finite-sliding, surface-to-surface contact pairs. Step cycling controls are specified for automatic calculation of the batch sizes and early termination of step cycling.
solid_sst_axi_wear.inp	Axisymmetric mesh with CAX4 elements. Used to generate a subsequent 3D model using symmetric model generation.
solid_sst_3D_wear.inp	A 3D hollow cylindrical mesh representative of a tire with C3D8R elements is generated using the revolve feature of symmetric model generation from the axisymmetric mesh in the previous input file. Contact is established initially with an analytic rigid surface representing the road. Wear is accumulated in a subsequent steady-state transport step. All the nodes along a streamline report the same wear distance magnitude accumulated in the steady-state transport step.

xpl_wear_curvedsurf_arig.inp	A 2D rigid surface is pressed into and slides back and forth in a hemispherical part modeled with CPS3 and CPS4R elements in Abaqus/Explicit. A wear model that has wear proportional to the friction coefficient is assigned to the deformable surface.
xpl_wear_curvedsurf_arig_ale.inp	A 2Drigid surface is pressed into and slides back and forth in a hemispherical part modeled with CPS3 and CPS4R elements in Abaqus/Explicit. A wear model that has wear proportional to the friction coefficient is assigned to the deformable surface. Arbitrary Lagrangian-Eulerian (ALE) Adaptive meshing is activated.
xpl_wear_slider.inp	A 2D block is pressed into and slides back and forth repetitively across a larger stationary block. The analysis is run using Abaqus/Explicit. A wear model that has wear proportional to the friction coefficient is assigned to the contact domain.
xpl_wear_slider_ale.inp	A 2D block is pressed into and slides back and forth repetitively across a larger stationary block. The analysis is run using Abaqus/Explicit. A wear model that has wear proportional to the friction coefficient is assigned to the contact domain. Arbitrary Lagrangian-Eulerian (ALE) adaptive meshing is activated.

Contact surface representation of threads

Products: Abaqus/Standard

References:

- *Contact Surface Representation for Threaded Interfaces*

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Features tested

Internally generated surface-element representation of thread geometry.

Problem description

Tests verify the behavior of automatically generated surface-element representation of thread geometry, which helps to improve contact simulation accuracy involving relatively large twisting motion between the external and the internal threads. The technique applies to general contact analyses for both finite-sliding and small-sliding.

Results and discussion

In all cases, the tests confirm that automatically generated surface elements can accurately represent thread geometry. The automatically generated surface elements provide detailed contact output. The solid elements of the threaded parts (rather than the surface elements) provide output variables such as stresses and strains at integration points.

Input files

surfElem_threads_tension.inp	Default threaded interface setting, finite-sliding.
surfElem_threads_tension_smsl.inp	Default threaded interface setting, small-sliding.
surfElem_threads_tension_leftHanded.inp	Left-handed threads, finite-sliding.
surfElem_threads_tension_linearFacet.inp	Thread representation using first-order surface elements, finite-sliding.
surfElem_threads_tension_externalExtent.inp	User-specified external threads extent, finite-sliding.
surfElem_threads_tension_internalExtent.inp	User-specified internal threads extent, finite-sliding.
surfElem_threads_tension_axis.inp	User-specified threads axis, finite-sliding.

Miscellanous

This section defines basic tests that provide evidence that the implementation of miscellaneous contact options provide the expected results.

In this section:

- *Contact between discrete points*
- *Automatic element conversion for surface contact*
- *Automatic surface definition and surface trimming*
- *Contact surface extensions*
- *Adjusting contact surface normals at symmetry planes*
- *Contact controls*
- *Multiple surface contact with penalty method*
- *Automated contact patch algorithm for finite-sliding deformable surfaces*

Contact between discrete points

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

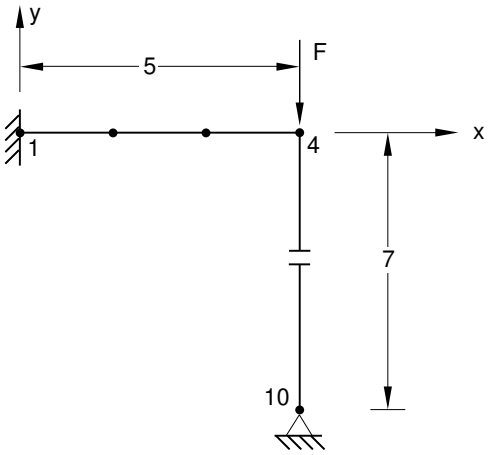
Elements tested

GAPUNI GAPCYL GAPSPHER

Problem description

Simple beam models are used to verify unidirectional, cylindrical, and spherical gap elements.

GAPUNI with positive gap clearance:



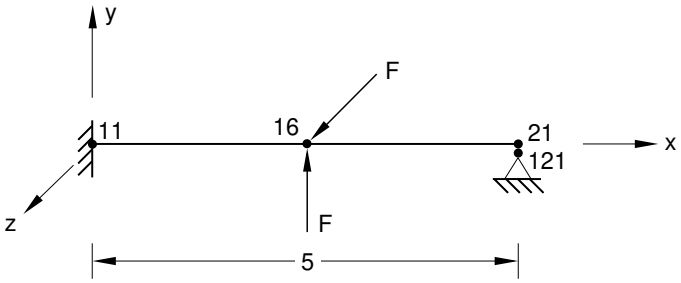
GAP data: Initial clearance = 0.5.

X, Y, Z direction cosine of the closure direction = (0., −1., 0.).

Boundary conditions: node 1 is clamped, and node 10 is fixed in the *x*- and *y*-directions.

Loading case 1: $F_y = -50$ at node 4; Loading case 2: $F_y = -100$ at node 4.

GAPCYL with positive gap clearance:



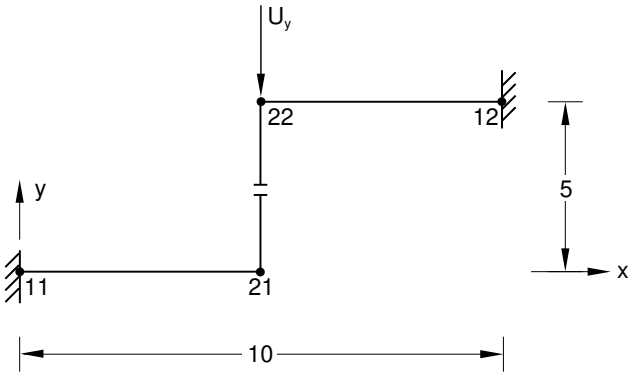
GAP data: Initial clearance = 0.0208 (positive gap clearance).

X, Y, Z direction cosine of the cylinder axis = (1., 0., 0.).

Boundary conditions: node 11 is clamped, node 121 is fixed in the *x*-, *y*- and *z*-directions.

Loading: Step 1: $F_y = 2.0 \times 10^4$ at node 16; Step 2: $F_z = 3.0 \times 10^4$ at node 16.

GAPCYL with negative gap clearance:

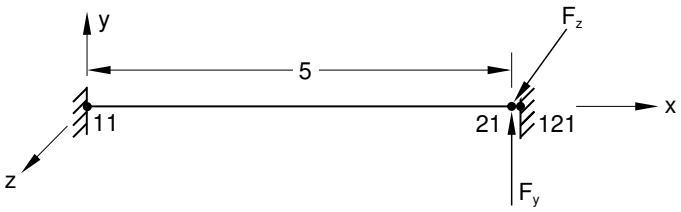


GAP data: Initial clearance = −1.0 (negative gap clearance).

X, Y, Z direction cosine of the cylinder axis = (1., 0., 0.).

Boundary conditions: nodes 11 and 12 are clamped, $u_y = -5.0$ at node 22.

GAPSPHER with positive gap clearance:



GAP data: Initial clearance = 0.2080.

Boundary conditions: nodes 11 and 121 are clamped.

Loading case 1: $F_y = 2.0 \times 10^4$ and $F_z = 3.0 \times 10^4$ at node 21; Loading case 2: $F_y = 4.0 \times 10^4$ and $F_z = 6.0 \times 10^4$ at node 21. The NLGEOM parameter is used.

Results and discussion

The contact constraints are satisfied properly.

Input files

eiu1sgcp.inp	GAPUNI element with positive gap clearance, perturbation step with **LOAD CASE*.
eic1sgcp.inp	GAPCYL element with positive gap clearance.
eic1sgcn.inp	GAPCYL element with negative gap clearance.
eis1sgcp.inp	GAPSPHER element with positive gap clearance, perturbation step with **LOAD CASE*.

Automatic element conversion for surface contact

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D15 C3D15V C3D20 C3D27 S8R5 S9R5

Features tested

Conversion of C3D20, C3D15, and S8R5 elements.

Problem description

These tests verify the automatic element conversion feature of Abaqus. With this feature if a quadratic element is specified as part of a secondary surface definition and there is no midface node on the contacting face, Abaqus automatically generates a midface node and modifies the element definition appropriately. Temperatures and predefined field variables at the automatically generated nodes are determined by interpolation from the existing, user-defined nodes. The conversion of C3D20, C3D15, and S8R5 elements into C3D27, C3D15V, and S9R5 elements is tested for the case of contact between a deformable body and a rigid surface, as well as contact between two deformable bodies.

In the first test a uniform temperature change of 50° is first applied to all of the elements to verify the temperature interpolation of the automatic conversion procedure. The elements then undergo uniform compression via contact with a frictionless rigid surface. The solution is compared to an identical model composed of C3D27, C3D15V, and S9R5 elements defined explicitly in the input file (no conversion is necessary). The second and third tests verify contact between pairs of deformable bodies in which the elements of the secondary surface undergo automatic conversion.

In all three cases the material is assumed elastic with Young's modulus of 3×10^6 lb/in^2, Poisson's ratio of 0.3, and a thermal expansion coefficient of 1×10^{-6}.

Results and discussion

In the first test all elements experience a uniform thermal strain of 5×10^{-5}. The results at the completion of Step 3 for the model with converted elements agree with the results for the model in which no elements undergo conversion.

Input files

ei39srsx.inp	Finite-sliding contact between deformable bodies and a rigid surface.
ei39sisx.inp	Small-sliding contact between two deformable bodies.
ei39sisx_surf.inp	Small-sliding contact between two deformable bodies, surface-to-surface constraint enforcement method.
ei39sfsx.inp	Finite-sliding contact between two deformable bodies.

Automatic surface definition and surface trimming

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D4 C3D8 CPS3 CPS4

Problem description

The input files *ele_trim2d.inp* and *ele_trim3d.inp* verify the automatic surface generation capability and trimming of surfaces. When a surface is defined without specifying the face identifiers of elements, the faces in the element set that are on the exterior (free) surface of the model form the surface. This definition may result in the inclusion of unwanted faces. Surface trimming provides the user with some basic control over the extent of open surfaces created on solid element meshes.

The input file *ele_trimdef.inp* tests the default trimming option. Abaqus will, by default, trim all contact surfaces except main surfaces involved in a finite-sliding contact pair.

Results and discussion

Some of the examples from the tests are shown below. They illustrate the recursive elimination of the ends of two-dimensional surfaces and the edges of three-dimensional surfaces. Trimming has no effect on closed surfaces (ones with no ends or edges). In each example the shaded elements in the model are used as the element set in the surface definition. The automatic surface generated and the surface generated by trimming are shown separately.

Trimming of two-dimensional surfaces

Figure 1 and *Figure 2* show how trimming of surfaces works for two-dimensional quadrilateral elements. Any face that includes an end node and a corner node is removed during trimming. *Figure 3* and *Figure 4* show trimming of surfaces for two-dimensional triangular elements.

Trimming of three-dimensional surfaces

Figure 5 and *Figure 6* show the trimming of surfaces for three-dimensional brick elements. *Figure 7* and *Figure 8* show how trimming of surfaces works for three-dimensional tetrahedron elements.

Default trimming of contact surfaces

The default trimming option was investigated for surfaces involved in small-sliding, finite-sliding, and both small- and finite-sliding contact pairs.

Input files

ele_trim2d.inp	Trimming of two-dimensional surfaces.
ele_trim3d.inp	Trimming of three-dimensional surfaces.

ele_trimdef.inp	Default trimming of contact surfaces.

Figures

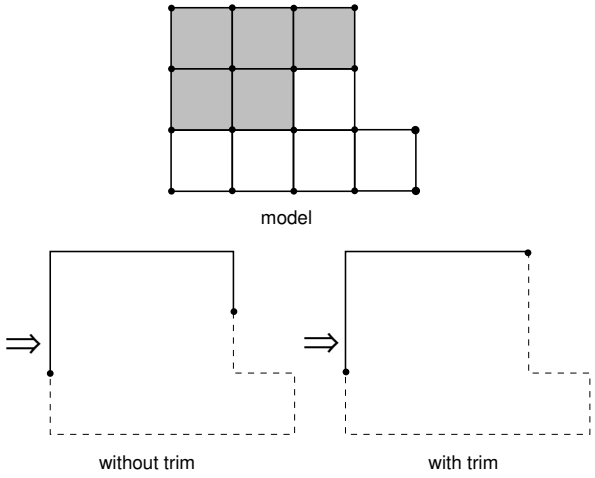


Figure 1: Quadrilateral elements—Example 1.

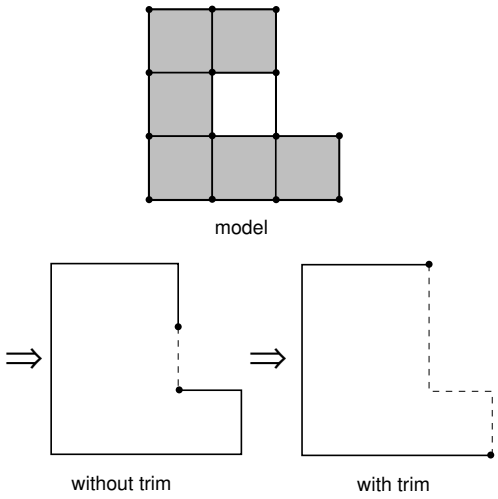


Figure 2: Quadrilateral elements—Example 2.

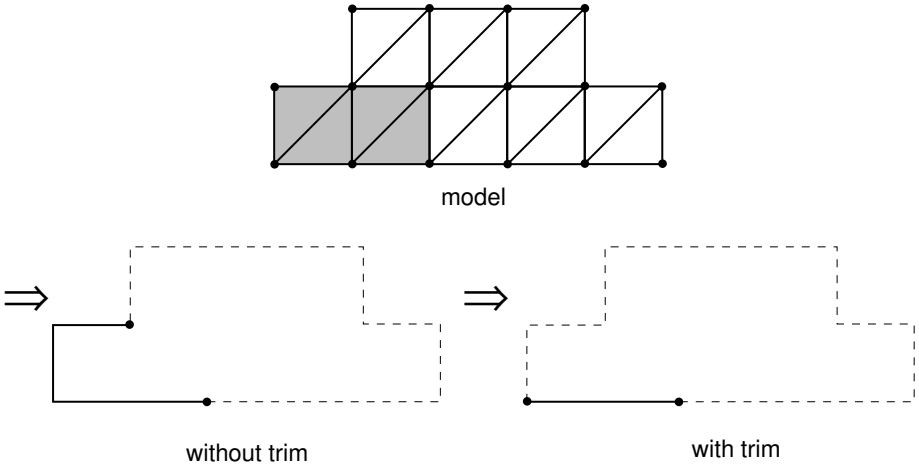


Figure 3: Triangular elements—Example 1.

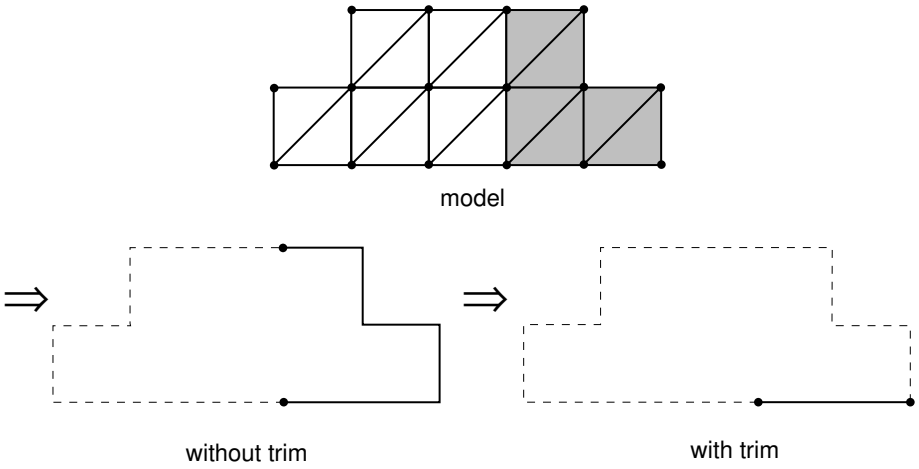


Figure 4: Triangular elements—Example 2.

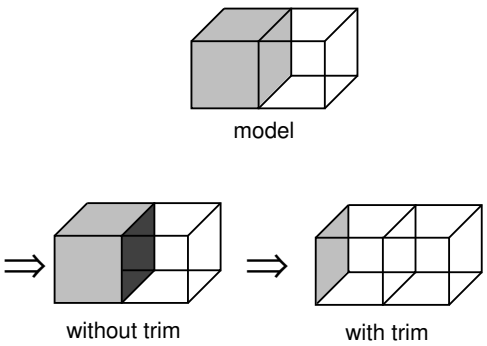


Figure 5: Brick elements—Example 1.

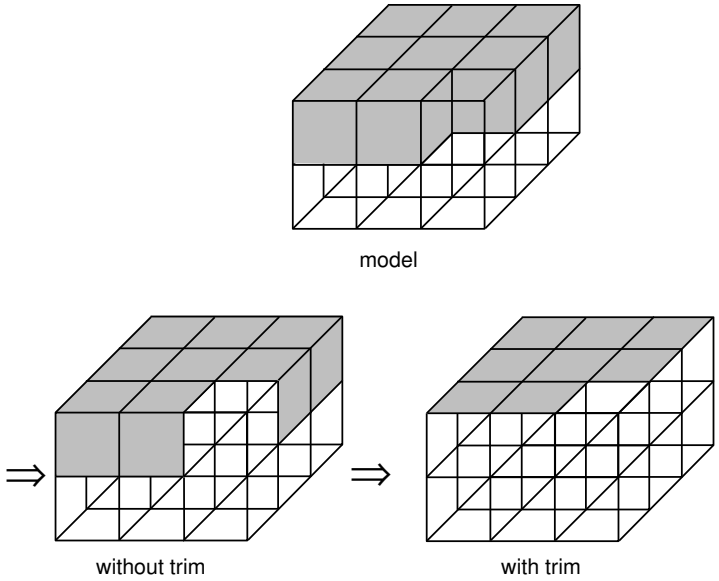


Figure 6: Brick elements—Example 2.

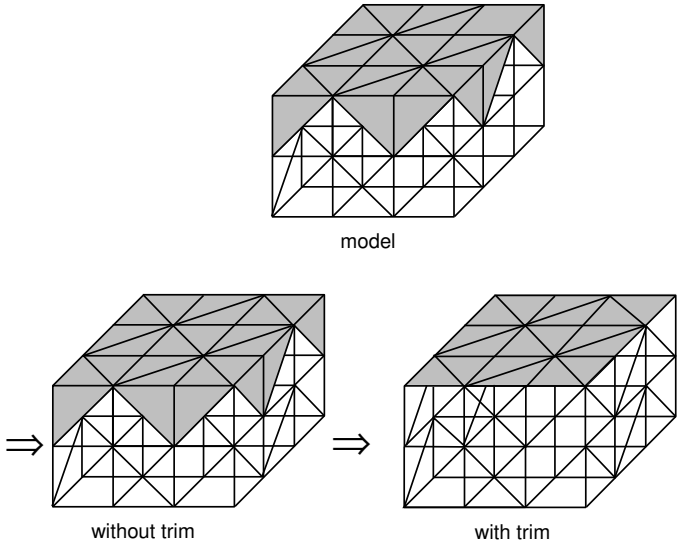


Figure 7: Tetrahedron elements—Example 1.

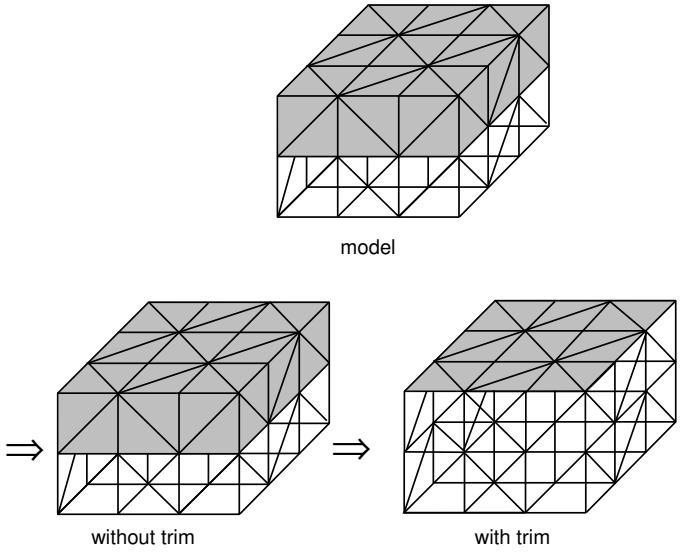


Figure 8: Tetrahedron elements—Example 2.

Contact surface extensions

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D8 C3D10 C3D10HS C3D20 CPE3 CPE4 CPE6 CPE8 ISL21A ISL22A

Features tested

Contact surface and slide line extensions for small- and finite-sliding.

Problem description

In small-sliding contact extending the main surface allows the secondary node to find an intersection with the main surface when the secondary node lies slightly outside the perimeter of the main surface at the start of the analysis. The small-sliding models consist of a stacked block arrangement in which the nodes of the secondary surface extend beyond the perimeter of the main surface at the start of the analysis.

In finite-sliding contact extending the main surface can prevent nodes from “falling-off” or getting trapped behind the main surface. The finite-sliding models are similar to the small-sliding models, except that the secondary surface lies within the perimeter of the main surface at the start of the analysis. A second step moves the secondary surface beyond the perimeter of the main surface but within the extension zone.

Material:

Young's modulus	3.0×10^6
Poisson's ratio	0.2

Results and discussion

The small-sliding tests verify that an intersection is found and that the proper contact clearance is calculated at the start of an analysis.

The contact clearances, slip distances, and contact pressures are used to verify the finite-sliding results when a secondary node enters the extension region.

Input files

ei38sfsx.inp	C3D8 elements, finite-sliding.
ei38sisx.inp	C3D8 elements, small-sliding.
ei38sisx_surf.inp	C3D8 elements, small-sliding, surface-to-surface constraint enforcement method.
ei3tsfsx.inp	C3D10 elements, finite-sliding.
ei3tsisx.inp	C3D10 elements, small-sliding.

ei3tsisx_surf.inp	C3D10 elements, small-sliding, surface-to-surface constraint enforcement method.
ei3tsfsx_c3d10hs.inp	C3D10HS elements, finite-sliding.
ei3tsisx_c3d10hs.inp	C3D10HS elements, small-sliding.
ei3tsisx_surf_c3d10hs.inp	C3D10HS elements, small-sliding, surface-to-surface constraint enforcement method.
ei3ssfsx.inp	C3D20 elements, finite-sliding.
ei3ssisx.inp	C3D20 elements, small-sliding.
ei3ssisx_surf.inp	C3D20 elements, small-sliding, surface-to-surface constraint enforcement method.
ei23sfsx.inp	CPE3 elements, finite-sliding.
ei23sisx.inp	CPE3 elements, small-sliding.
ei23sisx_surf.inp	CPE3 elements, small-sliding, surface-to-surface constraint enforcement method.
ei24sfsx.inp	CPE4 elements, finite-sliding.
ei24sisx.inp	CPE4 elements, small-sliding.
ei24sisx_surf.inp	CPE4 elements, small-sliding, surface-to-surface constraint enforcement method.
ei26sfsx.inp	CPE6 elements, finite-sliding.
ei26sisx.inp	CPE6 elements, small-sliding.
ei26sisx_surf.inp	CPE6 elements, small-sliding, surface-to-surface constraint enforcement method.
ei28sfsx.inp	CPE8 elements, finite-sliding.
ei28sisx.inp	CPE8 elements, small-sliding.
ei28sisx_surf.inp	CPE8 elements, small-sliding, surface-to-surface constraint enforcement method.
ei21sfix.inp	ISL21A elements.
ei22sfix.inp	ISL22A elements.

Adjusting contact surface normals at symmetry planes

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D8 C3D10M CPE3 CPE4 CPE8

Features tested

Contact surface normals are tested at symmetry planes for small- and finite-sliding contact.

Problem description

For small-sliding contact the tests verify that the surface normals are properly adjusted such that a secondary node finds an intersection with a curved main surface at the symmetry plane (see *Figure 1*). It also verifies that the proper clearance is calculated at the symmetry plane.

For finite-sliding contact the tests verify that the surface normals are properly adjusted and that the end segments of a two-dimensional contact surface are properly smoothed at the symmetry plane.

Some input files use a local nodal coordinate system to ensure that the surface normals are properly adjusted for the local system.

The models consist of two concentric deformable cylinders. A quarter-symmetry model is used. The initial clearance between both cylinders is 0.1. The loading consists of two steps. In the first step a pressure of 100 is applied on the outer cylinder such that the surface comes into contact with the inner cylinder. In the second step the pressure is released such that the elastic model returns to its original state.

Material:

Young's modulus	3.0×10^3
Poisson's ratio	0.2

Results and discussion

The clearances and contact pressures were verified analytically. The clearances for the finite-sliding test cases are slightly greater than the discretized clearance because of the smoothed main surface.

Input files

ei38sisn.inp	C3D8 elements, small-sliding.
ei38sisn_surf.inp	C3D8 elements, small-sliding, surface-to-surface constraint enforcement method.
ei3tsisn.inp	C3D10M elements, small-sliding.

ei3tsisn_surf.inp	C3D10M elements, small-sliding, surface-to-surface constraint enforcement method.
ei23sfsn.inp	CPE3 elements, finite-sliding.
ei23sisn.inp	CPE3 elements, small-sliding.
ei23sisn_surf.inp	CPE3 elements, small-sliding, surface-to-surface constraint enforcement method.
ei24sfsn.inp	CPE4 elements, finite-sliding.
ei24sisn.inp	CPE4 elements, small-sliding.
ei24sisn_surf.inp	CPE4 elements, small-sliding, surface-to-surface constraint enforcement method.
ei28sfsn.inp	CPE8 elements, finite-sliding.
ei28sfsn_auglagr.inp	CPE8 elements, finite-sliding.
ei28sisn.inp	CPE8 elements, small-sliding.
ei28sisn_surf.inp	CPE8 elements, small-sliding, surface-to-surface constraint enforcement method.
ei28sisn_auglagr.inp	CPE8 elements, small-sliding.
ei28sisn_auglagr_surf.inp	CPE8 elements, small-sliding, surface-to-surface constraint enforcement method.

Figures

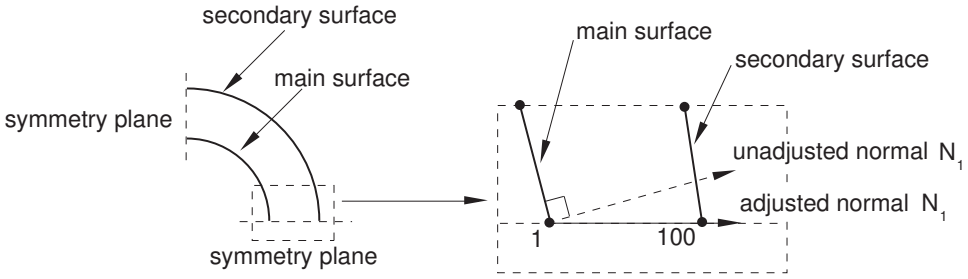


Figure 1: Adjusted normal at the symmetry plane.

Contact controls

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Contact stabilization

Elements tested

C3D8 CPE4

Features tested

Contact stabilization

Problem description

Contact stabilization can be used to control rigid body motions that may exist in a model before contact is fully developed. The option adds viscous damping in both the normal and tangential directions. By default, the damping is calculated automatically, but it is possible to modify the damping coefficient, the variation of the damping coefficient over the step, the range over which the damping works, and the ratio between normal and tangential damping. The controls specified with this option remain in effect until they are either changed by another contact controls procedure or reset to their default values. Contact stabilization can be defined for a specific contact pair or for the entire model. Further description of the stabilization controls can be found in *Generally Applicable Contact Controls in Abaqus/Standard*. In these tests various combinations of stabilization controls are tested in multistep analyses with multiple contact pairs.

The first group of analyses consists of six pairs of blocks that are pushed together in Step 1, subjected to tangential sliding in Step 2, and pulled apart in Step 3. The blocks are elastic, and the motion of the blocks is controlled with boundary conditions. Contact stabilization parameters are specified for the whole model and are overridden by different parameters for several individual contact pairs. The stabilization parameters vary from step to step. A restart file is written, and some restarts are made to test the restart functionality.

The second group of analyses consists of three blocks that are pushed together in Step 1, subjected to tangential sliding in Step 2, and pulled apart in Step 3. The blocks are elastic; and the top and bottom blocks are controlled with boundary conditions, whereas the middle block is completely free and held in place by contact stabilization. Different contact stabilization parameters are used for each contact pair. In addition, frictional properties are prescribed for one contact pair. This group contains two-dimensional and three-dimensional static analyses as well as a dynamic analysis.

Results and discussion

The results show contact damping pressures CDPRESS as well as contact damping shear stresses CDSHEAR1 and CDSHEAR2 that are in agreement with expectations. In addition, in the second group of problems the rigid body motions of the middle block are controlled and no solver messages are observed.

Input files

controlsstab_3d.inp	Static analysis with six pairs of blocks and different control parameters.
controlsstab_restart1.inp	Restart from the results of the analysis with six pairs of blocks.

controlsstab_restart2.inp	Restart from the results of the first restart analysis.
controlsstab_free_2d.inp	Static analysis with two fixed and one free block in two dimensions.
controlsstab_free_3d.inp	Static analysis with two fixed and one free block in three dimensions.
controlsstab_dyn.inp	Dynamic analysis with two fixed and one free block in three dimensions.

Tangential contact controls

Elements tested

C3D20R C3D27R

Features tested

Modifying the tangential penalty stiffness for all contact pairs in a linear perturbation step.

Problem description

During linear perturbation steps, all points in contact (i.e., with a “closed” status) are assumed to be sticking if friction is present. However, stick conditions are not enforced for contact nodes for which a velocity differential is imposed by the motion of the reference frame or the transport velocity. Stick conditions are enforced with a penalty method by default, and the perturbation tangent scale factor can be used to scale the penalty stiffness. For example, setting this parameter to zero will result in zero penalty stiffness, such that the stick conditions are not enforced during the perturbation step. Setting this parameter to a value greater than unity results in a larger-than-default penalty stiffness and, thus, stricter enforcement of stick conditions during the perturbation step.

The model consists of two blocks of different sizes in contact, with a nonzero friction coefficient in effect. In the first and second general steps we establish contact and apply a tangential displacement boundary condition such that the small block slips along the larger block. Natural frequencies are computed in subsequent perturbation steps:

Step Name	Perturbation Tangent Scale Factor
Frequency1	Not specified (default setting is 1.0)
Frequency2	Set to 1.0 (same as default)
Frequency3	Set to 0.0 (same as frictionless)
Frequency4	Set to 10^3

Material:

Young's modulus	2×10^7
Poisson's ratio	0.3
Friction coefficient	0.2

Results and discussion

Steps 3 and 4 (step names “Frequency1” and “Frequency2”) provide identical results, as expected. Step 5 (step name “Frequency3) has three zero-frequency eigenmodes corresponding to relative sliding between the two blocks, consistent with frictionless behavior. Strict enforcement of stick conditions is apparent in the eigenmodes for Step 6 (step name “Frequency4”).

Input files

pertbcntctrl.inp	Input file for this analysis.

Pressure-dependent contact controls

Elements tested

CPS4 C3D8

Features tested

Pressure-dependent contact constraint enforcement

Problem description

By default, during linear perturbation steps contact constraints are usually fully enforced for surfaces where contact is active independent of local normal pressure in the base state. This treatment of contact constraints may excessively stiffen the structure because certain areas of the model where the contact pressure is close to zero may actually come in and out of contact during vibrations. The nature of linear perturbation procedures does not allow such contact chattering. However, you can use contact pressure-dependent constraint enforcement to relax or even completely remove the constraints that have low pressure.

You can control the behavior through two user-specified contact pressure coefficients, p_0 and p_1; by default, p_0 is equal to p_1. For pressures greater than p_1, the constraints are fully enforced, as in the default contact constraint treatment. For pressures between p_0 and p_1 (if p_0 is less than p_1), the constraints are gradually weakened. For pressures less than p_0, the constraints are totally removed.

The model consists of a plane stress disk placed between two anvils. A sequence of linear perturbation frequency steps and general steps is applied. During the general steps the disk is compressed between the anvils, and the linear perturbation frequency steps compute eigenpairs with different parameter settings as shown in *Table 1*.

Table 1: Sequence of linear perturbation frequency and general steps.

Step Name	p_0	p_1
`Frequency1`	Not used	Not used
`General1`	Not used	Not used
`Frequency2`	Not used	Not used
`Frequency3`	0.	5×10^4
`Frequency4`	5×10^4	5×10^4
`Frequency5`	0.	5×10^4
`General2`	Not used	Not used
`Frequency6`	0.	0.
`Frequency7`	Not used	Not used
`Frequency8`	1×10^{15}	1×10^{15}

Material:

Young's modulus	2×10^5

Poisson's ratio	0.3
Friction coefficient	0.1

Results and discussion

During the `Frequency1` step minimum boundary conditions are imposed to compute the nonrigid body eigenpairs of the disk without any prestress. The `General1` step compresses the disk between the anvils, so contact constraints are established with a maximum contact pressure value of more than 6×10^4. The `Frequency2` step computes the eigenpairs of the stressed structure, and the `Frequency2` spectrum increases compared to `Frequency1`. In the `Frequency3` step p_0=0 and p_1=5×10^4, so some of the contact constraints are relaxed. Accordingly, the eigenvalues of `Frequency3` decrease compared to `Frequency2`. In `Frequency4` $p_0 = p_1 = 5 \times 10^4$, so some constraints are removed completely, and the spectrum decreases further. `Frequency5` has the same pressure-dependent constraint enforcement specification as `Frequency3` but uses a local contact pair instead of a global specification, and the results of `Frequency5` and `Frequency3` are the same. The `General2` step produces additional compression, so the eigenvalues of the next `Frequency6` linear perturbation step increase compared to `Frequency2`. The `Frequency6` p_0=p_1=0 results are coincident with the `Frequency7` default constraint enforcement results. In `Frequency8` both p_0 and p_1 are equal to a large number, so all contact constraints are removed. However, the `Frequency8` results are not coincident with the `Frequency1` results because of geometrically nonlinear effects in the loaded structure.

Two simple two-dimensional and three-dimensional two-element problems are also run using the same step sequence described above. The problems are geometrically linear, however, so the `Frequency8` results are the same as the `Frequency1` results.

Input files

pressdepper_disk.inp	Disk analysis.
pressdepper2d.inp	Two-dimensional two-element problem.
pressdepper3d.inp	Three-dimensional two-element problem.

Multiple surface contact with penalty method

Products: Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

S4R R3D4

Features tested

Three-dimensional penalty contact, accounting for penalty stiffness in the stable time increment, three-dimensional shell thickness in contact.

This problem tests the features listed but does not provide independent verification of the response.

Problem description

This example illustrates characteristics of penalty contact. The penalty method is a nondefault alternative to kinematic enforcement of contact constraints, and it is invoked by specifying the penalty method to enforce the contact constraints. In this example the penalty method is used to enforce contact between three bodies: a rigid plate, a rigid sphere, and an originally flat shell. The initial configuration is shown in *Figure 1*. The rigid plate is fully constrained. The rigid sphere is initially motionless. The initial velocity of the shell body causes the sphere to be pinched between the other two bodies, and deformation of the shell eventually leads to contact between the shell and the rigid plate.

An analytical rigid surface is used to model the rigid plate. An element-based rigid surface defined by R3D4 elements is used to model the rigid sphere. A deformable surface is defined over the shell body. Contact between each combination of these surfaces is defined with three contact pairs.

It would be preferable to model the sphere as an analytical surface, since the element-based surface is a nonsmooth approximation to the shape. However, analytical surfaces can act as main surfaces only, and this example requires the sphere to act as a secondary surface; therefore, the sphere must be modeled with elements. Element-based rigid surfaces can act as secondary surfaces with the penalty method, unlike with the kinematic contact method. This aspect of the penalty method allows contact modeling between rigid surfaces, such as between the rigid plate and the rigid sphere in this example. Having a rigid surface act, at least partially, as a secondary surface often will improve contact enforcement for rigid-to-deformable contact because nodes of a pure main surface can penetrate secondary facets without generating contact forces. In this example balanced main-secondary weighting is used for contact between the rigid sphere and the shell. If kinematic contact were used to model contact between the sphere and the shell, the sphere would have to be weighted as a pure main surface and the sphere nodes would be allowed to penetrate the shell facets.

It is generally preferable to use an analytical rigid surface whenever possible, rather than an element-based rigid surface, since an element-based approximation to a smooth surface can contribute to noise in a solution if secondary nodes from other surfaces slide across the element facets. However, this type of sliding is not significant in this problem.

Two sphere masses are considered for this example: 10^{-2} and 10^{-4}. The mass of the rigid sphere does not influence the deformation of the shell significantly, but this mass is significant with respect to numerical stability considerations. The maximum penalty stiffness allowed for numerical stability is directly proportional to the

contact mass and has a complex inverse dependence on the time increment. The contact mass corresponds approximately to the mass of the lighter rigid body or node of a deformable body involved in a contact constraint. Default penalty stiffnesses for contact involving one or two deformable surfaces are chosen to have a small effect (about 4% at most) on the element-by-element stable time increment for parent elements along the surface. The penalty stiffnesses that are chosen by default to enforce contact between rigid bodies do not influence the time increment. Hence, the default penalty stiffness will tend to decrease as the contact mass decreases.

Penalty stiffnesses can be modified by scaling the default values, which can influence the stable time increment. The stable time increment is affected by penalty contact only while the surfaces are in contact. A penalty scale factor of 10 has been specified for contact pairs involving the rigid sphere in the analysis with the lighter sphere, so we can expect that penalty contact will have a greater influence on the time incrementation in that analysis.

Results and discussion

The deformed configuration for the first analysis is shown in *Figure 2*. Contour plots of the vertical displacement of the shell for the two analyses are shown in *Figure 3* and *Figure 4*. The final shell configuration is nearly the same in the two models. These plots demonstrate that energy stored in penalty contact is recoverable, because shell nodes have rebounded after hitting the rigid plate. By default, viscous contact damping is activated for penalty contact, so a small amount of the energy stored in the penalty contact constraints is dissipated. This type of rebound would not occur if kinematic contact were used, since kinematic contact assumes "perfect plastic" impact.

History plots of the displacement of the rigid sphere for the two analyses are shown in *Figure 5*. The rigid sphere bounces back and forth between the other surfaces. The frequency of this oscillation is much higher for the analysis with the lighter sphere. Displacement of the rigid sphere exceeding 2.38×10^{-3} corresponds to penetration of the element-based rigid sphere into the rigid plate. For a smooth sphere of radius 10^{-2}, a displacement exceeding 2.0×10^{-3} would correspond to penetration. The penetration of the element-based sphere into the plate is plotted in *Figure 6*. The penetration is on the same order of magnitude for the two analyses. If the default penalty stiffnesses had been used for the analysis with the lighter sphere, the penetrations would have been an order of magnitude larger. In most analyses the contact penetrations will not be significant with the default penalty stiffnesses, but "pinching" of the sphere between the other two surfaces causes the penetration to be moderately significant in this example. Penetrations in a given problem can be reduced by increasing the penalty scale factor at a cost of decreasing the stable time increment.

The stable time increment is obtained based on an element-by-element estimate to demonstrate the effect of penalty contact on the stable time increment of the elements. History plots of the time increment for the two analyses are shown in *Figure 7*. For the analysis that uses the default penalty stiffnesses, the time increment dips by about 4% for increments in which the shell surface contacts either or both rigid surfaces. For the analysis with a specified penalty scale factor of 10, the time increment reductions associated with contact are more significant, as expected. In this case the time increment is cut by nearly a third in many increments in which the surfaces are in contact, and the number of increments for the analysis is nearly twice that of the analysis with the heavier sphere. When the penalty scale factor applies to contact pairs involving rigid surfaces, the time increment is reduced by roughly the square root of the penalty scale factor value during increments in which contact occurs. The effect of the penalty scale factor on the time increment is somewhat less significant for contact between deformable surfaces.

Input files

multpenaltycont1.inp	Analysis with the sphere mass equal to 10^{-2} and the time increment based on the element-by-element estimate.
multi1_gcont.inp	General contact analysis with the sphere mass equal to 10^{-2} and the time increment based on the element-by-element estimate.
multpenaltycont2.inp	Analysis with the sphere mass equal to 10^{-4} and the time increment based on the element-by-element estimate.

multi2_gcont.inp	General contact analysis with the sphere mass equal to 10^{-4} and the time increment based on the element-by-element estimate.
multpenaltycont3.inp	Analysis with the sphere mass equal to 10^{-2} and the time increment based on the global estimate.
multi3_gcont.inp	General contact analysis with the sphere mass equal to 10^{-2} and the time increment based on the global estimate.
multpenaltycont4.inp	Analysis with the sphere mass equal to 10^{-4} and the time increment based on the global estimate.
multi4_gcont.inp	General contact analysis with the sphere mass equal to 10^{-4} and the time increment based on the global estimate.
multpnltykincont.inp	Analysis testing both penalty and kinematic contact pairs.
multi_kin_gcont.inp	Analysis testing both general contact and kinematic contact pairs.
sphere_n.inp	External file containing the node data for these analyses.
sphere_e.inp	External file containing the element data for these analyses.

Figures

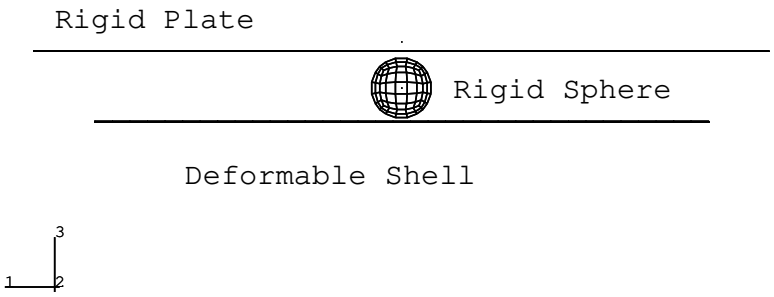


Figure 1: Initial configuration.

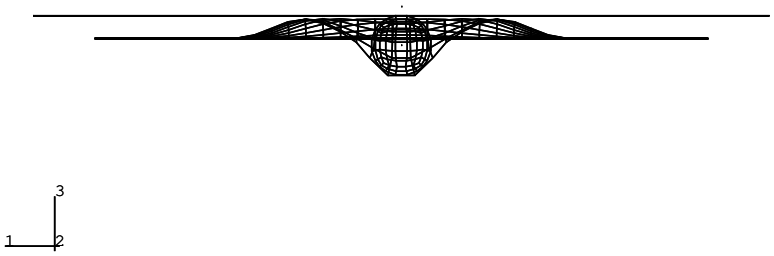


Figure 2: Final configuration.

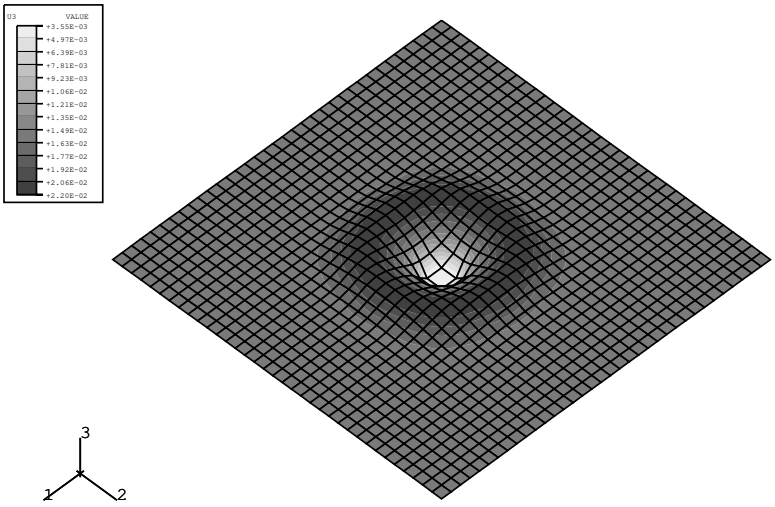


Figure 3: Deformed configuration of shell for analysis with larger sphere mass.

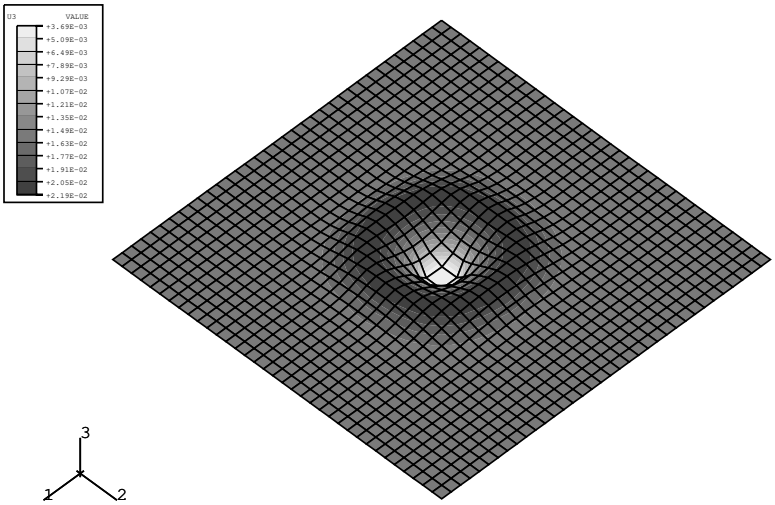


Figure 4: Deformed configuration of shell for analysis with smaller sphere mass.

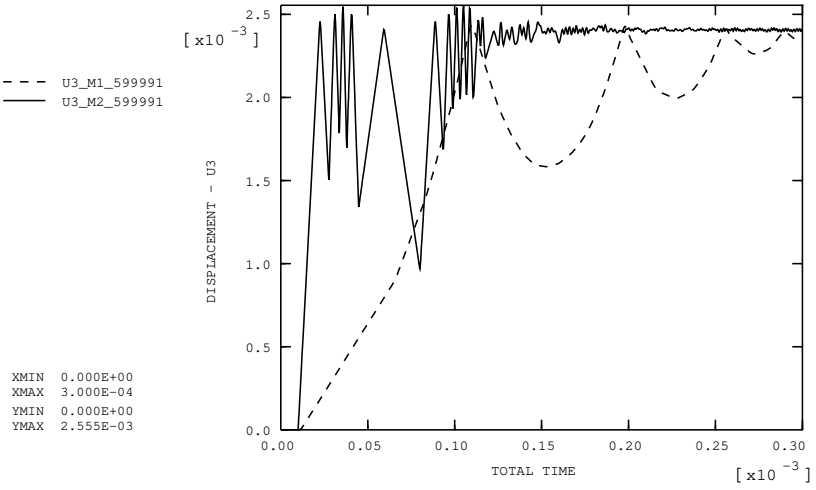


Figure 5: Sphere displacement versus time.

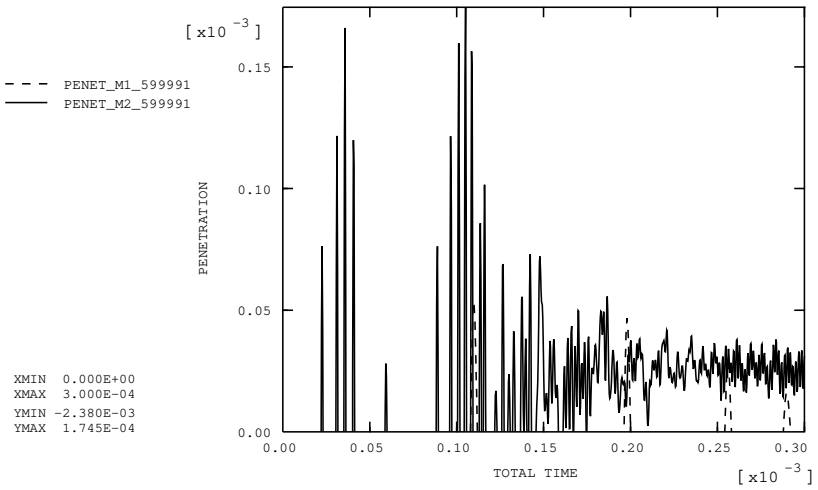


Figure 6: Penetration distance of sphere into rigid plate versus time.

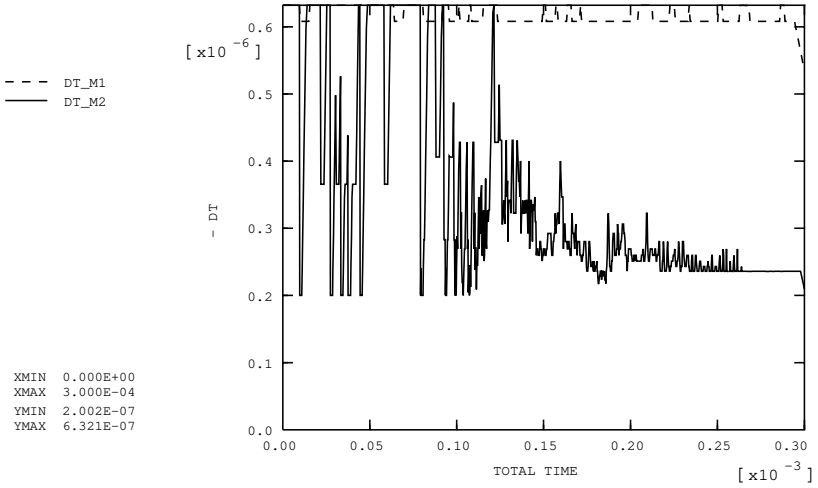


Figure 7: History of the time increment magnitude.

Automated contact patch algorithm for finite-sliding deformable surfaces

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D8 C3D10M SC8R CSS8

Features tested

The automatic contact patch and element reordering algorithm.

Problem description

These tests exercise the automatic contact patch and element reordering algorithm used to minimize the wavefront for three-dimensional deformable-to-deformable finite-sliding simulations.

Model: The model consists of a base block and two slider blocks resting on the base block. The dimension of the base block is $10 \times 6 \times 1$, and the dimension of each slider block is $1 \times 1 \times 1$. The model is illustrated in *Figure 1*.

Mesh: Two meshes are defined. The first mesh uses the 10-node modified tetrahedron, C3D10M, element; and the second mesh uses the 8-node solid, C3D8, element to define the base block. The base block consists of 300 C3D10M elements for the first mesh and 60 C3D8 elements for the second mesh. The slider block consists of four C3D8R elements. The main surface is defined on the top of the base block, and the secondary surface is defined on the bottom of each slider block. A total of 18 contact elements are generated by Abaqus.

Material: The following elastic properties are used:

Young's modulus	3×10^6
Poisson's ratio	0.0

Boundary conditions: The base block is fully restrained on the bottom. Contact is established in the first step by placing the slider blocks onto the base block with a prescribed boundary condition. A uniform pressure of 100 and 200 is applied to the slider blocks in the second step. The slider blocks are moved independently by prescribing a velocity in the subsequent steps.

Results and discussion

Contact stresses, element stresses in the slider blocks, and nodal displacements are verified. In addition, restart and post analysis jobs exist to verify that the correct analysis databases are accessed.

Input files

contactpatch_c3d10m.inp	C3D10M element test.

contactpatch_c3d10m_surf.inp	C3D10M element test using surface-to-surface contact.
contactpatch_c3d10m_restart.inp	Restart file for contactpatch_c3d10m.inp.
contactpatch_c3d10m_postoutput.inp	**POST OUTPUT* file for contactpatch_c3d10m.inp.
contactpatch_c3d8.inp	C3D8 element test.
contactpatch_c3d8_restart.inp	Restart file for contactpatch_c3d8.inp.
contactpatch_c3d8_postoutput.inp	**POST OUTPUT* file for contactpatch_c3d8.inp.
contactpatch_sc8r.inp	SC8R element test.
contactpatch_sc8r_restart.inp	Restart file for contactpatch_sc8r.inp.
contactpatch_sc8r_postoutput.inp	**POST OUTPUT* file for contactpatch_sc8r.inp.
contactpatch_css8.inp	CSS8 element test.

Figures

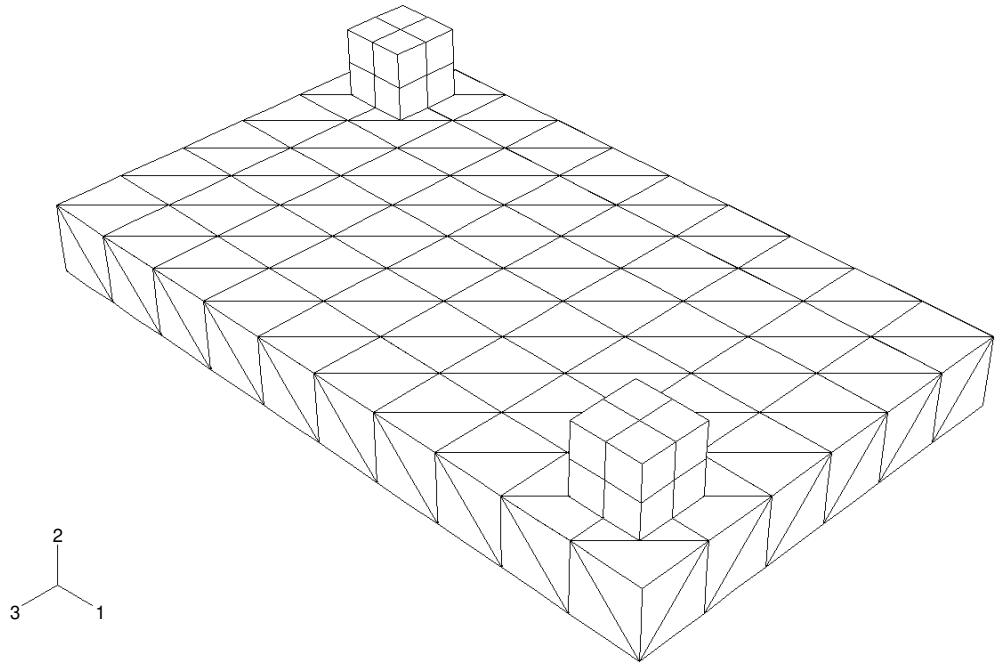


Figure 1: Model using the 10-node modified tetrahedral elements.

Materials Verification

This section defines the basic tests that are used to provide evidence that the implementation of the material options in Abaqus produces the expected results. Mechanical properties, thermal properties, and piezoresistivity properties are tested. For each mechanical material model listed, options and dependencies are exercised in stress/strain paths that are relevant to the particular material model. The material verification tests are also performed in all the different stress spaces available for each particular material model by choosing suitable finite elements.

In this section:

- *Mechanical properties*
- *Thermal properties*
- *Piezoresistivity properties: Piezoresistive effect*

Mechanical properties

In this section:

- *Elastic materials*
- *Viscoelastic materials*
- *Mullins effect and permanent set*
- *Hysteretic materials*
- *Temperature-dependent elastic materials*
- *Field-variable-dependent elastic materials*
- *Large-strain viscoelasticity with hyperelasticity*
- *Nonlinear large-strain viscoelasticity with hyperelasticity*
- *Transient internal pressure loading of a viscoelastic cylinder*
- *Rate-independent plasticity*
- *Rate-dependent plasticity in Abaqus/Standard*
- *Rate-dependent plasticity in Abaqus/Explicit*
- *Annealing temperature*
- *Temperature-dependent inelastic materials*
- *Field-variable-dependent inelastic materials*
- *Barlat anisotropic plasticity*
- *Johnson-Cook plasticity*
- *Porous metal plasticity*
- *Drucker-Prager plasticity*
- *Drucker-Prager/Cap plasticity model*
- *Paperboard material model*
- *Equation of state material*
- *Progressive damage and failure of ductile metals*
- *Progressive damage and failure in fiber-reinforced materials*
- *Generalized damage and failure for anisotropic materials*
- *Creep*
- *Concrete smeared cracking*
- *Concrete damaged plasticity*
- *Two-layer viscoplasticity*
- *Brittle cracking constitutive model*
- *Cracking model: tension shear test*
- *Hydrostatic fluid*
- *Composite, mass proportional, and rotary inertia proportional damping in Abaqus/Standard*
- *Material damping*
- *Mass proportional damping*
- *Structural damping*
- *Thermal expansion test in Abaqus/Standard*
- *Thermal expansion test in Abaqus/Explicit*
- *Field expansion test*
- *Multiscale materials*

- *Cure modeling in Abaqus/Standard*
- *Plasticity corrections*
- *Electrical and mechanical modeling of cardiac tissue*

Elastic materials

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Linear orthotropic elastic materials

Elements tested

C3D8 CPE4 CPS4

Problem description

Material:

Engineering constants		Stiffness coefficients	
E_1	1000.	D_{1111}	1000.
E_2	1000.	D_{1122}	0.
E_3	1000.	D_{2222}	1010.1
ν_{12}	0.	D_{1133}	0.
ν_{13}	0.	D_{2233}	101.01
ν_{23}	0.1	D_{3333}	1010.1
G_{12}	100.	D_{1212}	100.
G_{13}	100.	D_{1313}	100.
G_{23}	100.	D_{2323}	100.

Results and discussion

The results agree well with exact analytical or approximate solutions.

Input files

meloro3ltr.inp	**ELASTIC*, TYPE=ORTHOTROPIC; C3D8 elements.
meloro2ltr.inp	**ELASTIC*, TYPE=ORTHOTROPIC; CPS4 elements.
meleco3ltr.inp	**ELASTIC*, TYPE=ENGINEERING CONSTANTS; CPE4 elements.

Linear anisotropic elastic material

Elements tested

C3D8

Problem description

Material:

Stiffness coefficients	
D_{1111}	2.24×10^{11}
D_{1122}	4.79×10^{5}
D_{2222}	1.23×10^{11}
D_{1133}	4.21×10^{5}
D_{2233}	4.74×10^{5}
D_{3333}	1.21×10^{11}
D_{1112}	1×10^{6}
D_{2212}	2×10^{6}
D_{3312}	3×10^{6}
D_{1212}	7.69×10^{10}
D_{1113}	4×10^{6}
D_{2213}	5×10^{6}
D_{3313}	6×10^{6}
D_{1213}	7×10^{6}
D_{1313}	7.69×10^{10}
D_{1123}	8×10^{6}
D_{2223}	9×10^{6}
D_{3323}	10×10^{6}
D_{1223}	11×10^{6}
D_{1323}	12×10^{6}
D_{2323}	9×10^{9}

Results and discussion

The results agree well with exact analytical or approximate solutions.

Input files

melano3ltr.inp **ELASTIC*, TYPE=ANISOTROPIC; C3D8 elements.

Logarithmic porous elasticity

Elements tested
CAX8R

Problem description

Material:

Logarithmic bulk modulus, $\kappa = 1.0$

Poisson's ratio, $\nu = 0.3$

(The units are not important.)

Initial conditions:

Initial void ratio, $e_0 = 1.08$

Results and discussion

The results agree well with exact analytical or approximate solutions.

Input files

mpespo3ahc.inp Hydrostatic compression, CAX8R elements.

mpespo3vlp.inp Linear perturbation steps containing **LOAD CASE*, hydrostatic compression, CAX8R elements.

Power law–based porous elasticity

Elements tested
C3D8 , C3D8R

Problem description

Material:

$E_{ref} = 5000.0$

$p_{ref} = 3.0$

$p_0 = 1.0$

$n = 0.5$

$\nu_\infty = 0.3$

$\nu_0 = 0.3$

$m = 1.0$

(The units are not important.)

Results and discussion

The results agree well with exact analytical or approximate solutions.

Input files

porelapowlawtests.inp Hydrostatic compression, C3D8 and C3D8R elements.

Hypoelasticity

Elements tested

CPS4R

Problem description

Material: The following dependence of E on the second strain invariant I_2 is used:

E	ν	I_2
637.5	0.499	4.542×10^{-3}
700.3	0.499	1.6621×10^{-2}
765.7	0.499	3.4418×10^{-2}
840.7	0.499	5.6607×10^{-2}
917.4	0.499	8.2201×10^{-2}

Results and discussion

The results agree well with exact analytical or approximate solutions.

Input files

mhooto2hut.inp Nearly incompressible, uniaxial tension, CPS4R elements.

Hyperelasticity with polynomial strain energy function

Elements tested

C3D8RH CAX8 CGAX8H CPS4R

Problem description

Material:

Polynomial coefficients (N=1): $C_{10} = 80.$, $C_{01} = 20.$

Compressible case: $D_1 = 0.001$.

Test data (N=2): Treloar's experimental data.

(The units are not important.)

Results and discussion

The results agree well with exact analytical or approximate solutions.

Input files

Coefficient input

mhecoo3hut.inp	Incompressible, uniaxial tension, C3D8RH elements.
mhecoo3ibt.inp	Incompressible, biaxial tension, C3D8RH elements.
mhecoo3gsh.inp	Incompressible, planar tension, C3D8RH elements.
mhecoo3vlp.inp	Incompressible, uniaxial tension with static linear perturbation steps containing **LOAD CASE*, C3D8RH elements.
mhecot3hut.inp	Incompressible, temperature-dependent, uniaxial tension, C3D8RH elements.
mhecdo3hut.inp	Compressible, uniaxial tension, C3D8RH elements.
mhecdo3ibt.inp	Compressible, biaxial tension, C3D8RH elements.
mhecdo3gsh.inp	Compressible, planar tension, C3D8RH elements.
mhecdo3ahc.inp	Compressible, volumetric compression, C3D8RH elements.
mhecoo2hut.inp	Incompressible, uniaxial tension, CPS4R elements.
mhecoo2ibt.inp	Incompressible, biaxial tension, CPS4R elements.
mhecoo2gsh.inp	Incompressible, planar tension, CPS4R elements.
mhecdo2hut.inp	Compressible, uniaxial tension, CPS4R elements.
mhecdo2ibt.inp	Compressible, biaxial tension, CPS4R elements.
mhecdo2gsh.inp	Compressible, planar tension, CPS4R elements.
mhecoo2spt.inp	Incompressible, pure torsion, CGAX8H elements.
mhecoo2eit.inp	Incompressible; extension, inflation, and torsion; CGAX8H elements.
mhecoo2eis.inp	Incompressible; extension, inflation, and shear; CAX8 elements.

Test data input

mhetdo3hut.inp	Compressible, uniaxial tension, C3D8RH elements.
mhetdo3ibt.inp	Compressible, biaxial tension, C3D8RH elements.
mhetdo3gsh.inp	Compressible, planar tension, C3D8RH elements.
mhetdo3ahc.inp	Compressible, volumetric compression, C3D8RH elements.
mhetdi3ahc.inp	Compressible, volumetric compression with initial stresses, C3D8RH elements.

Hyperelasticity with reduced polynomial strain energy function

Elements tested

C3D8RH CPS4R

Problem description

Material:

Polynomial coefficients (N=1): C_{10} = 100.

Compressible case: D_1 = 0.001.

Test data (N=6): Treloar's experimental data.

(The units are not important.)

Results and discussion

The results agree well with exact analytical or approximate solutions.

Input files

Coefficient input

mhrcoo3hut.inp	Incompressible, uniaxial tension, C3D8RH elements.
mhrcoo3ibt.inp	Incompressible, biaxial tension, C3D8RH elements.
mhrcoo3gsh.inp	Incompressible, planar tension, C3D8RH elements.
mhrcoo3vlp.inp	Incompressible, uniaxial tension with static linear perturbation steps containing **LOAD CASE*, C3D8RH elements.
mhrcot3hut.inp	Incompressible, temperature-dependent, uniaxial tension, C3D8RH elements.
mhrcdo3hut.inp	Compressible, uniaxial tension, C3D8RH elements.
mhrcdo3ibt.inp	Compressible, biaxial tension, C3D8RH elements.
mhrcdo3gsh.inp	Compressible, planar tension, C3D8RH elements.
mhrcdo3ahc.inp	Compressible, volumetric compression, C3D8RH elements.
mhrcoo2hut.inp	Incompressible, uniaxial tension, CPS4R elements.
mhrcoo2ibt.inp	Incompressible, biaxial tension, CPS4R elements.
mhrcoo2gsh.inp	Incompressible, planar tension, CPS4R elements.
mhrcdo2hut.inp	Compressible, uniaxial tension, CPS4R elements.
mhrcdo2ibt.inp	Compressible, biaxial tension, CPS4R elements.
mhrcdo2gsh.inp	Compressible, planar tension, CPS4R elements.

Test data input

mhrtdo3hut.inp	Compressible, uniaxial tension, C3D8RH elements.
mhrtdo3ibt.inp	Compressible, biaxial tension, C3D8RH elements.
mhrtdo3gsh.inp	Compressible, planar tension, C3D8RH elements.

mhrtdo3ahc.inp	Compressible, volumetric compression, C3D8RH elements.

Hyperelasticity with neo-Hookean strain energy function

Elements tested

C3D8RH CPS4R

Problem description

Material:

Neo-Hookean coefficient: C_{10} = 100.

Compressible case: D_1 = 0.001.

Test data: Treloar's experimental data.

(The units are not important.)

Results and discussion

The results agree well with exact analytical or approximate solutions.

Input files

Coefficient input

mhncoo3hut.inp	Incompressible, uniaxial tension, C3D8RH elements.
mhncoo3ibt.inp	Incompressible, biaxial tension, C3D8RH elements.
mhncoo3gsh.inp	Incompressible, planar tension, C3D8RH elements.
mhncoo3vlp.inp	Incompressible, uniaxial tension with static linear perturbation steps containing **LOAD CASE*, C3D8RH elements.
mhncot3hut.inp	Incompressible, temperature-dependent, uniaxial tension, C3D8RH elements.
mhncdo3hut.inp	Compressible, uniaxial tension, C3D8RH elements.
mhncdo3ibt.inp	Compressible, biaxial tension, C3D8RH elements.
mhncdo3gsh.inp	Compressible, planar tension, C3D8RH elements.
mhncdo3ahc.inp	Compressible, volumetric compression, C3D8RH elements.
mhncoo2hut.inp	Incompressible, uniaxial tension, CPS4R elements.
mhncoo2ibt.inp	Incompressible, biaxial tension, CPS4R elements.
mhncoo2gsh.inp	Incompressible, planar tension, CPS4R elements.
mhncdo2hut.inp	Compressible, uniaxial tension, CPS4R elements.
mhncdo2ibt.inp	Compressible, biaxial tension, CPS4R elements.
mhncdo2gsh.inp	Compressible, planar tension, CPS4R elements.

Test data input

mhntdo3hut.inp	Compressible, uniaxial tension, C3D8RH elements.
mhntdo3ibt.inp	Compressible, biaxial tension, C3D8RH elements.

mhntdo3gsh.inp	Compressible, planar tension, C3D8RH elements.
mhntdo3ahc.inp	Compressible, volumetric compression, C3D8RH elements.

Hyperelasticity with Mooney-Rivlin strain energy function

Elements tested

C3D8RH CPS4R

Problem description

Material:

Mooney-Rivlin coefficients: C_{10} = 80., C_{01} = 20.

Compressible case: D_1 = 0.001.

Test data: Treloar's experimental data.

(The units are not important.)

Results and discussion

The results agree well with exact analytical or approximate solutions.

Input files

Coefficient input

mhmcoo3hut.inp	Incompressible, uniaxial tension, C3D8RH elements.
mhmcoo3ibt.inp	Incompressible, biaxial tension, C3D8RH elements.
mhmcoo3gsh.inp	Incompressible, planar tension, C3D8RH elements.
mhmcoo3vlp.inp	Incompressible, uniaxial tension with static linear perturbation steps containing **LOAD CASE*, C3D8RH elements.
mhmcot3hut.inp	Incompressible, temperature-dependent, uniaxial tension, C3D8RH elements.
mhmcdo3hut.inp	Compressible, uniaxial tension, C3D8RH elements.
mhmcdo3ibt.inp	Compressible, biaxial tension, C3D8RH elements.
mhmcdo3gsh.inp	Compressible, planar tension, C3D8RH elements.
mhmcdo3ahc.inp	Compressible, volumetric compression, C3D8RH elements.
mhmcoo2hut.inp	Incompressible, uniaxial tension, CPS4R elements.
mhmcoo2ibt.inp	Incompressible, biaxial tension, CPS4R elements.
mhmcoo2gsh.inp	Incompressible, planar tension, CPS4R elements.
mhmcdo2hut.inp	Compressible, uniaxial tension, CPS4R elements.
mhmcdo2ibt.inp	Compressible, biaxial tension, CPS4R elements.
mhmcdo2gsh.inp	Compressible, planar tension, CPS4R elements.

Test data input

mhmtdo3hut.inp	Compressible, uniaxial tension, C3D8RH elements.

mhmtdo3ibt.inp	Compressible, biaxial tension, C3D8RH elements.
mhmtdo3gsh.inp	Compressible, planar tension, C3D8RH elements.
mhmtdo3ahc.inp	Compressible, volumetric compression, C3D8RH elements.

Hyperelasticity with Yeoh strain energy function

Elements tested

C3D8RH CPS4R

Problem description

Material:

Yeoh coefficients: C_{10} = 100., C_{20} = −1., C_{30} = 0.01.

Compressible case: D_1 = 0.001.

Test data: Treloar's experimental data.

(The units are not important.)

Results and discussion

The results agree well with exact analytical or approximate solutions.

Input files

Coefficient input

mhycoo3hut.inp	Incompressible, uniaxial tension, C3D8RH elements.
mhycoo3ibt.inp	Incompressible, biaxial tension, C3D8RH elements.
mhycoo3gsh.inp	Incompressible, planar tension, C3D8RH elements.
mhycoo3vlp.inp	Incompressible, uniaxial tension with static linear perturbation steps containing **LOAD CASE*, C3D8RH elements.
mhycot3hut.inp	Incompressible, temperature-dependent, uniaxial tension, C3D8RH elements.
mhycdo3hut.inp	Compressible, uniaxial tension, C3D8RH elements.
mhycdo3ibt.inp	Compressible, biaxial tension, C3D8RH elements.
mhycdo3gsh.inp	Compressible, planar tension, C3D8RH elements.
mhycdo3ahc.inp	Compressible, volumetric compression, C3D8RH elements.
mhycoo2hut.inp	Incompressible, uniaxial tension, CPS4R elements.
mhycoo2ibt.inp	Incompressible, biaxial tension, CPS4R elements.
mhycoo2gsh.inp	Incompressible, planar tension, CPS4R elements.
mhycdo2hut.inp	Compressible, uniaxial tension, CPS4R elements.
mhycdo2ibt.inp	Compressible, biaxial tension, CPS4R elements.
mhycdo2gsh.inp	Compressible, planar tension, CPS4R elements.

Test data input

mhytdo3hut.inp	Compressible, uniaxial tension, C3D8RH elements.
mhytdo3ibt.inp	Compressible, biaxial tension, C3D8RH elements.
mhytdo3gsh.inp	Compressible, planar tension, C3D8RH elements.
mhytdo3ahc.inp	Compressible, volumetric compression, C3D8RH elements.

Hyperelasticity with Ogden strain energy function

Elements tested

C3D8RH CAX8 CGAX8H CPS4R

Problem description

Material:

Ogden coefficients (N=2): $\mu_1 = 160.$, $\alpha_1 = 2.$, $\mu_2 = 40.$, $\alpha_2 = -2.$

Compressible case: $D_1 = 0.001$.

Test data (N=2): Treloar's experimental data.

(The units are not important.)

Results and discussion

The results agree well with exact analytical or approximate solutions.

Input files

Coefficient input

mhgcoo3hut.inp	Incompressible, uniaxial tension, C3D8RH elements.
mhgcoo3ibt.inp	Incompressible, biaxial tension, C3D8RH elements.
mhgcoo3gsh.inp	Incompressible, planar tension, C3D8RH elements.
mhgcoo3vlp.inp	Incompressible, uniaxial tension with static linear perturbation steps containing **LOAD CASE*, C3D8RH elements.
mhgcot3hut.inp	Incompressible, temperature-dependent, uniaxial tension, C3D8RH elements.
mhgcdo3hut.inp	Compressible, uniaxial tension, C3D8RH elements.
mhgcdo3ibt.inp	Compressible, biaxial tension, C3D8RH elements.
mhgcdo3gsh.inp	Compressible, planar tension, C3D8RH elements.
mhgcdo3ahc.inp	Compressible, volumetric compression, C3D8RH elements.
mhgcoo2hut.inp	Incompressible, uniaxial tension, CPS4R elements.
mhgcoo2ibt.inp	Incompressible, biaxial tension, CPS4R elements.
mhgcoo2gsh.inp	Incompressible, planar tension, CPS4R elements.
mhgcdo2hut.inp	Compressible, uniaxial tension, CPS4R elements.
mhgcdo2ibt.inp	Compressible, biaxial tension, CPS4R elements.
mhgcdo2gsh.inp	Compressible, planar tension, CPS4R elements.

mhgcoo2spt.inp	Incompressible, pure torsion, CGAX8H elements.
mhgcoo2eit.inp	Incompressible; extension, inflation, and torsion; CGAX8H elements.
mhgcoo2eis.inp	Incompressible; extension, inflation, and shear; CAX8 elements.

Test data input

mhgtdo3hut.inp	Compressible, uniaxial tension, C3D8RH elements.
mhgtdo3ibt.inp	Compressible, biaxial tension, C3D8RH elements.
mhgtdo3gsh.inp	Compressible, planar tension, C3D8RH elements.
mhgtdo3ahc.inp	Compressible, volumetric compression, C3D8RH elements.

Hyperelasticity with Arruda-Boyce strain energy function

Elements tested

C3D8RH CPS4R

Problem description

Material:

Arruda-Boyce coefficients: $\mu = 200.$, $\lambda_m = 5.$

Compressible case: $D_1 = 0.001$.

Test data: Treloar's experimental data.

(The units are not important.)

Results and discussion

The results agree well with exact analytical or approximate solutions.

Input files

Coefficient input

mhacoo3hut.inp	Incompressible, uniaxial tension, C3D8RH elements.
mhacoo3ibt.inp	Incompressible, biaxial tension, C3D8RH elements.
mhacoo3gsh.inp	Incompressible, planar tension, C3D8RH elements.
mhacoo3vlp.inp	Incompressible, uniaxial tension with static linear perturbation steps containing **LOAD CASE*, C3D8RH elements.
mhacot3hut.inp	Incompressible, temperature-dependent, uniaxial tension, C3D8RH elements.
mhacdo3hut.inp	Compressible, uniaxial tension, C3D8RH elements.
mhacdo3ibt.inp	Compressible, biaxial tension, C3D8RH elements.
mhacdo3gsh.inp	Compressible, planar tension, C3D8RH elements.
mhacdo3ahc.inp	Compressible, volumetric compression, C3D8RH elements.
mhacoo2hut.inp	Incompressible, uniaxial tension, CPS4R elements.
mhacoo2ibt.inp	Incompressible, biaxial tension, CPS4R elements.

mhacoo2gsh.inp	Incompressible, planar tension, CPS4R elements.
mhacdo2hut.inp	Compressible, uniaxial tension, CPS4R elements.
mhacdo2ibt.inp	Compressible, biaxial tension, CPS4R elements.
mhacdo2gsh.inp	Compressible, planar tension, CPS4R elements.

Test data input

mhatdo3hut.inp	Compressible, uniaxial tension, C3D8RH elements.
mhatdo3ibt.inp	Compressible, biaxial tension, C3D8RH elements.
mhatdo3gsh.inp	Compressible, planar tension, C3D8RH elements.
mhatdo3ahc.inp	Compressible, volumetric compression, C3D8RH elements.

Hyperelasticity with Van der Waals strain energy function

Elements tested

C3D8RH CPS4R

Problem description

Material:

Van der Waals coefficients: $\mu = 200.$, $\lambda_m = 10.$, $a = 0.1$, $\beta = 0.$

Compressible case: $D_1 = 0.001$.

Test data: Treloar's experimental data.

Test data (parameter β held constant): Treloar's experimental data, $\beta = 0.$

(The units are not important.)

Results and discussion

The results agree well with exact analytical or approximate solutions.

Input files

Coefficient input

mhvcoo3hut.inp	Incompressible, uniaxial tension, C3D8RH elements.
mhvcoo3ibt.inp	Incompressible, biaxial tension, C3D8RH elements.
mhvcoo3gsh.inp	Incompressible, planar tension, C3D8RH elements.
mhvcoo3vlp.inp	Incompressible, uniaxial tension with static linear perturbation steps containing **LOAD CASE*, C3D8RH elements.
mhvcot3hut.inp	Incompressible, temperature-dependent, uniaxial tension, C3D8RH elements.
mhvcdo3hut.inp	Compressible, uniaxial tension, C3D8RH elements.
mhvcdo3ibt.inp	Compressible, biaxial tension, C3D8RH elements.
mhvcdo3gsh.inp	Compressible, planar tension, C3D8RH elements.
mhvcdo3ahc.inp	Compressible, volumetric compression, C3D8RH elements.

mhvcoo2hut.inp	Incompressible, uniaxial tension, CPS4R elements.
mhvcoo2ibt.inp	Incompressible, biaxial tension, CPS4R elements.
mhvcoo2gsh.inp	Incompressible, planar tension, CPS4R elements.
mhvcdo2hut.inp	Compressible, uniaxial tension, CPS4R elements.
mhvcdo2ibt.inp	Compressible, biaxial tension, CPS4R elements.
mhvcdo2gsh.inp	Compressible, planar tension, CPS4R elements.

Test data input

mhvtdo3hut.inp	Compressible, uniaxial tension, C3D8RH elements.
mhvtdo3ibt.inp	Compressible, biaxial tension, C3D8RH elements.
mhvtdo3gsh.inp	Compressible, planar tension, C3D8RH elements.
mhvtdo3ahc.inp	Compressible, volumetric compression, C3D8RH elements.

Test data input (parameter held constant)

mhvtbo3hut.inp	Compressible, uniaxial tension, C3D8RH elements.
mhvtbo3ibt.inp	Compressible, biaxial tension, C3D8RH elements.
mhvtbo3gsh.inp	Compressible, planar tension, C3D8RH elements.
mhvtbo3ahc.inp	Compressible, volumetric compression, C3D8RH elements.

Hyperelasticity with Marlow strain energy function

Elements tested

C3D8H C3D8R CPE4RH CPE4R CPS4R M3D4R S4R SC8R
T2D2 T3D2 B21 B22 B31 B32 B31OS B32OS
PIPE21 PIPE22 PIPE31 PIPE32

Problem description

The tests in this section verify that the results generated using the Marlow hyperelastic model with different elements agree with the test data specified in the model.

Results and discussion

The results agree well with the test data specified for the Marlow model.

Input files

Abaqus/Standard input files

marlow_uniaxial_icmp.inp	Incompressible, C3D8H elements, uniaxial tension test data.
marlow_uniaxial_cmp.inp	Compressible, C3D8H elements, uniaxial tension and volumetric compression test data.
marlow_uniaxial_pos.inp	Compressible, C3D8H elements, uniaxial tension test data, with Poisson's ratio equal to 0.47.
marlow_uniaxial_e3.inp	Compressible, C3D8H elements, uniaxial tension test data, with lateral nominal strains specified.
marlow_biaxial_icmp.inp	Incompressible, C3D8H elements, uniaxial tension test data.

marlow_biaxial_cmp.inp	Compressible, C3D8H elements, biaxial tension and volumetric compression test data.
marlow_biaxial_pos.inp	Compressible, C3D8H elements, biaxial tension test data, with Poisson's ratio equal to 0.47.
marlow_biaxial_eb.inp	Compressible, C3D8H elements, biaxial tension test data, with lateral nominal strains specified.
marlow_planar_icmp.inp	Incompressible, C3D8H elements, planar tension test data.
marlow_planar_cmp.inp	Compressible, C3D8H elements, planar tension and volumetric compression test data.
marlow_planar_pos.inp	Compressible, C3D8H elements, planar tension test data, with Poisson's ratio equal to 0.47.
marlow_planar_e3.inp	Compressible, C3D8H elements, planar tension test data, with lateral nominal strains specified.
marlow_cpe4rh_icmp.inp	Incompressible, CPE4RH elements, uniaxial tension test data.
marlow_cpe4rh_cmp.inp	Compressible, CPE4RH elements, uniaxial tension and volumetric compression test data.
marlow_cps4r_icmp.inp	Incompressible, CPS4R elements, uniaxial tension test data.
marlow_cps4r_cmp.inp	Compressible, CPS4R elements, uniaxial tension and volumetric compression test data.
marlow_s4r_icmp.inp	Incompressible, S4R elements, uniaxial tension test data.
marlow_s4r_cmp.inp	Compressible, S4R elements, uniaxial tension and volumetric compression test data.
marlow_sc8r_cmp.inp	Compressible, SC8R elements, uniaxial tension and volumetric compression test data.
marlow_m3d4r_icmp.inp	Incompressible, M3D4R elements, uniaxial tension test data.
marlow_m3d4r_cmp.inp	Compressible, M3D4R elements, uniaxial tension and volumetric compression test data.
marlow_t2d2_icmp.inp	Incompressible, T2D2 elements, uniaxial tension test data.
marlow_t2d2_cmp.inp	Compressible, T2D2 elements, uniaxial tension and volumetric compression test data.
marlow_t3d2_icmp.inp	Incompressible, T3D2 elements, uniaxial tension test data.
marlow_t3d2_cmp.inp	Compressible, T3D2 elements, uniaxial tension and volumetric compression test data.
marlow_b21_icmp.inp	Incompressible, B21 elements, uniaxial tension test data.
marlow_b21_cmp.inp	Compressible, B21 elements, uniaxial tension and volumetric compression test data.
marlow_b22_icmp.inp	Incompressible, B22 elements, uniaxial tension test data.
marlow_b22_cmp.inp	Compressible, B22 elements, uniaxial tension and volumetric compression test data.
marlow_b31_icmp.inp	Incompressible, B31 elements, uniaxial tension test data.
marlow_b31_cmp.inp	Compressible, B31 elements, uniaxial tension and volumetric compression test data.
marlow_b32_icmp.inp	Incompressible, B32 elements, uniaxial tension test data.
marlow_b32_cmp.inp	Compressible, B32 elements, uniaxial tension and volumetric compression test data.
marlow_b31os_icmp.inp	Incompressible, B31OS elements, uniaxial tension test data.

marlow_b31os_cmp.inp	Compressible, B31OS elements, uniaxial tension and volumetric compression test data.
marlow_b32os_icmp.inp	Incompressible, B32OS elements, uniaxial tension test data.
marlow_b32os_cmp.inp	Compressible, B32OS elements, uniaxial tension and volumetric compression test data.
marlow_pipe21_icmp.inp	Incompressible, PIPE21 elements, uniaxial tension test data.
marlow_pipe21_cmp.inp	Compressible, PIPE21 elements, uniaxial tension and volumetric compression test data.
marlow_pipe22_icmp.inp	Incompressible, PIPE22 elements, uniaxial tension test data.
marlow_pipe22_cmp.inp	Compressible, PIPE22 elements, uniaxial tension and volumetric compression test data.
marlow_pipe31_icmp.inp	Incompressible, PIPE31 elements, uniaxial tension test data.
marlow_pipe31_cmp.inp	Compressible, PIPE31 elements, uniaxial tension and volumetric compression test data.
marlow_pipe32_icmp.inp	Incompressible, PIPE32 elements, uniaxial tension test data.
marlow_pipe32_cmp.inp	Compressible, PIPE32 elements, uniaxial tension and volumetric compression test data.
marlow_combined_ct.inp	Incompressible, T2D2 elements, uniaxial tension and compression test data.
marlow_rebar.inp	Incompressible, S4R elements, uniaxial tension test data.
marlow_depend.inp	Incompressible, C3D8H elements, uniaxial tension test data with field dependencies.
marlow_hysteresis.inp	Incompressible, C3D8H elements, uniaxial tension test data, hysteresis analysis.
marlow_initialstress.inp	Compressible, C3D8H elements, uniaxial tension test data, with initial stress specified.
marlow_map0.inp	Incompressible, C3D8H elements, uniaxial tension test data, needed for the following map solution analysis.
marlow_map.inp	Incompressible, C3D8H elements, uniaxial tension test data, map solution analysis.
marlow_visco_c3d8h_icmp.inp	Compressible, C3D8H elements, uniaxial tension and relaxation test data.
marlow_visco_c3d8h_cmp.inp	Compressible; C3D8H elements; uniaxial tension, volumetric compression, and relaxation test data.
marlow_visco_cps4r_icmp.inp	Compressible, CPS4R elements, uniaxial tension and relaxation test data.
marlow_visco_cps4r_cmp.inp	Compressible; CPS4R elements; uniaxial tension, volumetric compression, and relaxation test data.
marlow_visco_t3d2_icmp.inp	Compressible, T3D2 elements, uniaxial tension and relaxation test data.
marlow_visco_t3d2_cmp.inp	Compressible; T3D2 elements; uniaxial tension, volumetric compression, and relaxation test data.
marlow_visco_b21_icmp.inp	Compressible, B21 elements, uniaxial tension and relaxation test data.
marlow_visco_b21_cmp.inp	Compressible; B21 elements; uniaxial tension, volumetric compression, and relaxation test data.
marlow_visco_b31_icmp.inp	Compressible, B31 elements, uniaxial tension and relaxation test data.

marlow_visco_b31_cmp.inp	Compressible; B32 elements; uniaxial tension, volumetric compression, and relaxation test data.
marlow_visco_pipe21_icmp.inp	Compressible, PIPE21 elements, uniaxial tension and relaxation test data.
marlow_visco_pipe21_cmp.inp	Compressible; PIPE21 elements; uniaxial tension, volumetric compression, and relaxation test data.
marlow_visco_pipe31_icmp.inp	Compressible, PIPE31 elements, uniaxial tension and relaxation test data.
marlow_visco_pipe31_cmp.inp	Compressible; PIPE31 elements; uniaxial tension, volumetric compression, and relaxation test data.
marlow_volumecomp.inp	Volumetric test, C3D8R elements, uniaxial tension and volumetric compression.
marlow_sx_s_c3d8r.inp	Base problem for carrying out import from Abaqus/Standard to Abaqus/Explicit, C3D8R element.
marlow_xs_s_c3d8r.inp	Import into Abaqus/Standard from `marlow_sx_x_c3d8r.inp`.
marlow_sx_s_cpe4r.inp	Base problem for carrying out import from Abaqus/Standard to Abaqus/Explicit, CPE4R elements.
marlow_xs_s_cpe4r.inp	Import into Abaqus/Standard from `marlow_sx_x_cpe4r.inp`.
marlow_sx_s_cps4r.inp	Base problem for carrying out import from Abaqus/Standard to Abaqus/Explicit, CPS4R elements.
marlow_xs_s_cps4r.inp	Import into Abaqus/Standard from `marlow_sx_x_cps4r.inp`.
marlow_sx_s_sc8r.inp	Base problem for carrying out import from Abaqus/Standard to Abaqus/Explicit, SC8R elements.
marlow_xs_s_sc8r.inp	Import into Abaqus/Standard from `marlow_sx_x_sc8r.inp`.
marlow_sx_s_t2d2.inp	Base problem for carrying out import from Abaqus/Standard to Abaqus/Explicit, T2D2 elements.
marlow_xs_s_t2d2.inp	Import into Abaqus/Standard from `marlow_sx_x_t2d2.inp`.

Abaqus/Explicit input files

marlow_xpl_disp.inp	Uniaxial test; displacement control; C3D8R, CPE4R, CPS4R, M3D4R, S4R, B31, and T3D2 elements.
marlow_xpl_load.inp	Uniaxial test; load control; C3D8R, CPE4R, CPS4R, M3D4R, S4R, B31, and T3D2 elements.
marlow_xpl_initstress.inp	Specified initial stress; C3D8R, CPE4R, CPS4R, M3D4R, S4R, B31, and T3D2 elements.
marlow_xpl_mullins.inp	Combination with **MULLINS EFFECT*; cyclic test; C3D8R, CPE4R, CPS4R, M3D4R, S4R, B31, and T3D2 elements.
marlow_xpl_visco.inp	Creep test; C3D8R, CPE4R, CPS4R, M3D4R, S4R, B31, and T3D2 elements.
marlow_sx_x_c3d8r.inp	Explicit dynamic continuation of `marlow_sx_s_c3d8r.inp`.
marlow_sx_x_cpe4r.inp	Explicit dynamic continuation of `marlow_sx_s_cpe4r.inp`.
marlow_sx_x_cps4r.inp	Explicit dynamic continuation of `marlow_sx_s_cps4r.inp`.
marlow_sx_x_sc8r.inp	Explicit dynamic continuation of `marlow_sx_s_sc8r.inp`.
marlow_sx_x_t2d2.inp	Explicit dynamic continuation of `marlow_sx_s_t2d2.inp`.

Hyperelasticity with Valanis-Landel strain energy function

Elements tested

C3D8 C3D8R C3D8RH CGAX8H CPE4 CPS4R M3D4R S4R T3D2

Problem description

The tests in this section verify that the results generated using different elements with the Valanis-Landel hyperelastic model agree with the test data specified in the model.

Results and discussion

The results agree well with the test data specified for the Valanis-Landel model.

Input files

Abaqus/Standard input files with uniaxial test data

valanis_landel_incmp_uni_c3d8rh.inp	Incompressible material, uniaxial tension, C3D8RH element.
valanis_landel_incmp_uni_cgax8h.inp	Incompressible material, torsion, CGAX8H element.

Abaqus/Standard input files with uniaxial and lateral strain test data

valanis_landel_cmp_uni_uniten_cpe4.inp	Compressible material, uniaxial tension, CPE4 element.
valanis_landel_cmp_uni_cps4r.inp	Compressible material, planar tension, CPS4R element.
valanis_landel_cmp_uni_uniten_cps4r.inp	Compressible material, uniaxial tension, CPS4R element.

Abaqus/Standard input files with uniaxial and volumetric test data

valanis_landel_cmp_univol_c3d8.inp	Compressible material, volumetric compression, C3D8 element.
valanis_landel_cmp_univol_c3d8rh.inp	Compressible material, volumetric compression, C3D8RH element.

Abaqus/Standard input files with uniaxial, biaxial, and volumetric test data

valanis_landel_incmp_uni_biaxial_c3d8rh.inp	Incompressible material, uniaxial tension, uniaxial and biaxial test data, C3D8RH element.
valanis_landel_cmp_uni_biaxial_c3d8rh.inp	Compressible material, uniaxial tension, uniaxial and biaxial test data, C3D8RH element.
valanis_landel_incmp_uni_biaxial_cgax8h.inp	Incompressible material, torsion, uniaxial and biaxial test data, CGAX8H element.
valanis_landel_cmp_unibiavol_c3d8.inp	Compressible material, uniaxial and biaxial test data, volumetric compression, C3D8 element.

Abaqus/Explicit input files

xpl_valanis_landel_c3d8_vol_both.inp	Tension test with both uniaxial and volumetric test data, displacement control, C3D8 element.
xpl_valanis_landel_c3d8_vol_unibiavol.inp	Tension test with uniaxial, biaxial, and volumetric test data; displacement control, C3D8 element.
xpl_valanis_landel_c3d8_vol_comp.inp	Tension test with both uniaxial and compression volumetric test data, displacement control, C3D8 element.

xpl_valanis_landel_c3d8_vol_unibiavolcomp.inp	Tension test with uniaxial, biaxial, and compression volumetric test data; displacement control; C3D8 element.
xpl_valanis_landel_he_c3d8_ic.inp	Uniaxial tension test with initial stress (both uniaxial and lateral strain test data), displacement control, C3D8 element.
xpl_valanis_landel_he_s4r.inp	Uniaxial tension test with both uniaxial and lateral strain test data, displacement control, S4R element.
xpl_valanis_landel_ve_c3d8_vol.inp	Tension test with viscoelasticity (both uniaxial and volumetric test data), displacement control, C3D8 element.
xpl_valanis_landel_ve_cps4r.inp	Tension test with viscoelasticity (both uniaxial and lateral strain test data), displacement control, CPS4R element.
xpl_valanis_landel_ve_t3d2_vol.inp	Tension test with viscoelasticity (both uniaxial and volumetric test data), displacement control, T3D2 element.
xpl_valanis_landel_fefp_c3d8_vol_both.inp	Tension test with plasticity (both uniaxial and volumetric test data), displacement control, C3D8 element.
xpl_valanis_landel_fefp_c3d8_ic.inp	Tension test with plasticity and initial stress (both uniaxial and lateral strain test data), displacement control, C3D8 element.
xpl_valanis_landel_fefp_3d_1d.inp	Tension test with plasticity (both uniaxial and volumetric test data), displacement control, C3D8R and T3D2 elements.
xpl_valanis_landel_fefp_m3d4r.inp	Tension test with plasticity (both uniaxial and volumetric test data), displacement control, M3D4R element.
prf_network_stiff_xpl_vl.inp	Tension test with nonlinear viscoelasticity (both uniaxial and lateral strain test data), displacement control, C3D8R element.
prf_network_stiff_xpl_vl_temp_fv.inp	Tension test with nonlinear viscoelasticity (both uniaxial and lateral strain test data), displacement control, temperature and field variable dependencies, C3D8R element.
valanis_landel_sx_s_c3d8r.inp	Base job for import analysis of `valanis_landel_sx_x_c3d8r.inp`.
valanis_landel_sx_x_c3d8r.inp	Explicit dynamic continuation of `valanis_landel_sx_s_c3d8r.inp`.
valanis_landel_xs_s_c3d8r.inp	Dynamic continuation of `valanis_landel_sx_x_c3d8r.inp`.
valanis_landel_xx_x_c3d8r.inp	Explicit dynamic continuation of `valanis_landel_sx_x_c3d8r.inp`.

Hyperelasticity with Hencky strain energy function

Elements tested

B31 C3D8 C3D8R C3D8H CAX4 CPE4 CPS4 M3D4R S4R T3D2

Problem description

Material:

Hencky coefficients	
Young's modulus	E=20.0
Poisson's ratio (compressible case)	ν=0.3
Poisson's ratio (incompressible case)	ν=0.5

(The units are not important.)

Results and discussion

The results agree well with exact analytical or approximate solutions.

Input files

Abaqus/Standard input files

hencky_b31.inp	Compressible material, uniaxial tension, B31 element.
hencky_shear_c3d8.inp	Compressible material, shear deformation, C3D8 element.
hencky_uniaxial_c3d8.inp	Compressible material, uniaxial tension, C3D8 element.
hencky_shear_c3d8h.inp	Incompressible material, shear deformation, C3D8H element.
hencky_uniaxial_cax4.inp	Compressible material, uniaxial tension, CAX4 element.
hencky_uniaxial_cpe4.inp	Compressible material, uniaxial tension, CPE4 element.
hencky_uniaxial_cps4.inp	Compressible material, uniaxial tension, CPS4 element.
hencky_uniaxial_m3d4r.inp	Compressible material, uniaxial tension, M3D4R element.

Abaqus/Explicit input files

xpl_hencky_shear_c3d8.inp	Shear deformation, C3D8 element.
xpl_hencky_he_c3d8r_ic.inp	Uniaxial tension test with initial stress, C3D8R element.
xpl_hencky_he_s4r.inp	Uniaxial tension test, S4R element.
xpl_hencky_ve_c3d8.inp	Uniaxial tension test with viscoelasticity, C3D8 element.
xpl_hencky_ve_t3d2.inp	Uniaxial tension test with viscoelasticity, T3D2 element.
hencky_xx_x1_c3d8r.inp	Base job for import analysis of `hencky_xx_x2_c3d8r.inp`.
hencky_xx_x2_c3d8r.inp	Explicit dynamic continuation of `hencky_xx_x1_c3d8r.inp`.
xpl_hencky_fefp_c3d8.inp	Tension test with plasticity, C3D8 element.
xpl_hencky_fefp_m3d4r.inp	Tension test with plasticity, M3D4R element.
xpl_hencky_fefp_c3d8_ic.inp	Tension test with plasticity and initial stress, C3D8 element.
prf_network_stiff_xpl_hencky.inp	Tension test with nonlinear viscoelasticity, C3D8R element.
prf_network_stiff_xpl_hencky_temp.inp	Tension test with nonlinear viscoelasticity, temperature dependencies, C3D8R element.
hencky_sx_s_c3d8r.inp	Base job for import analysis of `hencky_sx_x_c3d8r.inp`.
hencky_sx_x_c3d8r.inp	Explicit dynamic continuation of `hencky_sx_s_c3d8r.inp`.
hencky_xs_s_c3d8r.inp	Dynamic continuation of `hencky_sx_x_c3d8r.inp`.
hencky_xx_x_c3d8r.inp	Explicit dynamic continuation of `hencky_sx_x_c3d8r.inp`.

Hyperfoam

Elements tested

C3D8R CPS4R

Problem description

Material:

Hyperfoam coefficients (N=3, from fit of test data):

μ_1 = –48.3291, α_1 = 3.58961, μ_2 = 26.3505, α_2 = 3.84360, μ_3 = 22.1809, α_3 = 3.34171.

Test data (N=3): Uniaxial compression, simple shear test data.

An effective Poisson's ratio of $\nu_i = 0$ is used, except for $\nu_i = 0.10$ for the biaxial test cases and varying ν_i in the temperature-dependent case.

(The units are not important.)

Results and discussion

The results agree well with exact analytical or approximate solutions.

Input files

Coefficient input

mhfcdo2euc.inp	ν_i = 0., uniaxial compression, CPS4R elements.
mhfcdo2fbc.inp	ν_i = 0.1, biaxial compression, CPS4R elements.
mhfcdo2gsh.inp	ν_i = 0., simple shear, CPS4R elements.
mhfcdo3vlp.inp	ν_i = 0., uniaxial compression with linear perturbation steps containing **LOAD CASE*, C3D8R elements.

Test data input

mhftdo3euc.inp	ν_i = 0., uniaxial compression, C3D8R elements.
mhftdo3fbc.inp	ν_i = 0.1, biaxial compression, C3D8R elements.
mhftdo3gsh.inp	ν_i = 0., simple shear, C3D8R elements.
mhftdo3ahc.inp	ν_i = 0., volumetric compression, C3D8R elements.
mhfcdt3euc.inp	ν_i = 0. – 0.06, temperature-dependent, uniaxial compression, C3D8R elements.
mhftdi3ahc.inp	ν_i = 0., volumetric compression with initial stresses, C3D8R elements.

Low-density foam

Elements tested

C3D8R CPE4R T3D2

Problem description

The tests in this section verify that the results generated using the low-density foam model with different elements agree with the test data specified in the model.

Results and discussion

The results agree well with the rate-dependent test data specified for the low-density foam model.

Input files

Low-density foam with zero Poisson's ratio

lowdensfoam_uni.inp	Uniaxial compression with varying deformation rates; C3D8R, CPE4R, and T3D2 elements.
lowdensfoam_shr.inp	Simple shear test, C3D8R and CPE4R elements.
lowdensfoam_inistress.inp	Specified initial stress; C3D8R, CPE4R, and T3D2 elements.
xx_x1_lowdensfoam.inp	Base problem for carrying out import from Abaqus/Explicit to Abaqus/Explicit; uniaxial cyclic test; displacement control; C3D8R, CPE4R, and T3D2 elements.
xx_x2_lowdensfoam_n_y.inp	Continuation of `xx_x1_lowdensfoam.inp` with state imported.

Low-density foam with Poisson effects

lowdensfoam_poisson_uni.inp	Uniaxial compression with varying deformation rates, C3D8R and CPE4R elements.
lowdensfoam_poisson_shr.inp	Simple shear test, C3D8R and CPE4R elements.
lowdensfoam_poisson_inistress.inp	Specified initial stress, C3D8R and CPE4R elements.
xx_x1_lowdensfoam_poisson.inp	Base problem for carrying out import from Abaqus/Explicit to Abaqus/Explicit, uniaxial cyclic test, displacement control, C3D8R and CPE4R elements.
xx_x2_lowdensfoam_poisson_n_y.inp	Continuation of `xx_x1_lowdensfoam_poisson.inp` with state imported.

Anisotropic hyperelasticity with generalized Fung strain energy function

Elements tested

C3D8 C3D8R CPE4R CPS4R M3D4R S4R

Problem description

Material:

Fung coefficients	
c	26.95×10^3
b_{1111}	0.9925
b_{1122}	0.0749
b_{2222}	0.4180
b_{1133}	0.0295
b_{2233}	0.0193

Fung coefficients	
b_{3333}	0.0089
b_{1212}	5.0
b_{1313}	5.0
b_{2323}	5.0
Compressible case	
D=1.5 × 10^{-7} or D=1 × 10^{-8}	

(The units are not important.)

Results and discussion

The results agree well with exact analytical or approximate solutions.

Input files

Abaqus/Standard input files

uaniso_inv_fung.inp — Uniaxial tension; built-in Fung orthotropic model verified through user subroutines *UANISOHYPER_INV* and *UANISOHYPER_STRAIN*; load control; C3D8 element.

uanisohyper_inv.f — User subroutines *UANISOHYPER_INV* and *UANISOHYPER_STRAIN* to define the Fung orthotropic model.

funganiso_mullins_ve.inp — Uniaxial tension, loading-unloading; Fung anisotropic model with Mullins effect and viscoelasticity; C3D8R, CPE4R, and CPS4R elements.

Abaqus/Explicit input files

fung_disp_xpl.inp — Uniaxial cyclic test; displacement control; C3D8R, CPE4R, CPS4R, M3D4R, and S4R elements.

fung_load_xpl.inp — Uniaxial cyclic test; load control; C3D8R, CPE4R, CPS4R, M3D4R, and S4R elements.

fung_visco_xpl.inp — Combination with viscoelastic and Mullins effect; uniaxial cyclic test; C3D8R, CPE4R, CPS4R, M3D4R, and S4R elements.

xx_x1_fung_disp.inp — Base problem for carrying out import from Abaqus/Explicit to Abaqus/Explicit; uniaxial cyclic test; displacement control; C3D8R, CPE4R, CPS4R, M3D4R, and S4R elements.

xx_x2_fung_disp_n_y.inp — Continuation of `xx_x1_fung_disp.inp` with state imported.

xx_x1_fung_load.inp — Base problem for carrying out import from Abaqus/Explicit to Abaqus/Explicit; uniaxial cyclic test; load control; C3D8R, CPE4R, CPS4R, M3D4R, and S4R elements.

xx_x2_fung_load_n_y.inp — Continuation of `xx_x1_fung_load.inp` with state imported.

xx_x1_fung_visco.inp — Base problem for carrying out import from Abaqus/Explicit to Abaqus/Explicit; combination with viscoelastic and Mullins effect; uniaxial cyclic test; C3D8R, CPE4R, CPS4R, M3D4R, and S4R elements.

xx_x2_fung_visco_n_y.inp — Continuation of `xx_x1_fung_visco.inp` with state imported.

Anisotropic hyperelasticity with Holzapfel-Gasser-Ogden strain energy function

Elements tested

C3D8R C3D10 C3D10HS CPE4R CPS4R M3D4R S4R

Problem description

Material:

Holzapfel-Gasser-Ogden coefficients:

C_{10} = 7.64., k_1 = 996.6, k_2 = 524.6, κ = 0.226.

Fiber directions (N=2):

$$A_1 = (\cos\gamma, \sin\gamma, 0),$$

$$A_2 = (\cos\gamma, -\sin\gamma, 0),$$

with γ=49.98°.

Compressible case: $D = 1 \times 10^{-6}$.

(The units are not important.)

Results and discussion

The results agree well with exact analytical or approximate solutions.

Input files

Abaqus/Standard input files

hgo_2fiber_std_uni.inp	Uniaxial tension; displacement control; C3D8, C3D20R, and C3D10 elements.
hgo_2fiber_std_uni_c3d10hs.inp	Uniaxial tension; displacement control; C3D8, C3D20R, and C3D10HS elements.
hgo_2fiber_uni_hybrid.inp	Uniaxial tension; displacement control; C3D8H, C3D20RH, and C3D10H elements.
hgo_2fiber_std_uniori.inp	Uniaxial tension; displacement control; orientation along nonglobal Cartesian system; C3D8, C3D20R, and C3D10 elements.
hgo_2fiber_std_uniori_c3d10hs.inp	Uniaxial tension; displacement control; orientation along nonglobal Cartesian system; C3D8, C3D20R, and C3D10HS elements.
hgo_3fiber_std_uni.inp	Uniaxial tension, displacement control, C3D8 elements.
hgo_3fiber_uni_c3d8h.inp	Uniaxial tension, displacement control, C3D8H elements.
hgo_c3d8_std_ss.inp	Simple shear, displacement control, C3D8 elements.
hgo_c3d8_std_uni.inp	Cyclic uniaxial tension, displacement control, C3D8 elements.
hgo_2fiber_ehgc.inp	Simple shear; displacement control; C3D8, C3D8R, and CPE4R elements.
hgo_2fiber_pless.inp	Simple shear; displacement control; C3D8H, C3D8RH, and CPE4RH elements.
hgo_2fiber_ps.inp	Uniaxial tension; plane stress; load control; C3D8RH, CPS4R, M3D4R, and S4R elements.

hzplaniso_ve.inp	Viscoelastic behavior included; uniaxial tension, loading and unloading; C3D8R, CPE4R, and CPS4R elements.
uaniso_inv_hgople.inp	Plane strain tension/compression; user subroutine *UANISOHYPER_INV* verified using built-in Holzapfel strain energy function; displacement control; C3D8, CPE8R, and CPEG4 elements.
uaniso_inv_isople.inp	Plane strain tension; user subroutine *UANISOHYPER_INV* verified using built-in isotropic hyperelasticity; displacement control; C3D8H and CPE4H elements.

Abaqus/Explicit input files

holzapfel_disp_xpl.inp	Uniaxial cyclic test; displacement control; C3D8R, CPE4R, CPS4R, M3D4R, and S4R elements.
holzapfel_load_xpl.inp	Uniaxial cyclic test; load control; C3D8R, CPE4R, CPS4R, M3D4R, and S4R elements.
holzapfel_visco_xpl.inp	Combination with viscoelastic and Mullins effect; uniaxial cyclic test; C3D8R, CPE4R, CPS4R, M3D4R, and S4R elements.
xx_x1_holzapfel_disp.inp	Base problem for carrying out import from Abaqus/Explicit to Abaqus/Explicit, uniaxial cyclic test; displacement control; C3D8R, CPE4R, CPS4R, M3D4R, and S4R elements.
xx_x2_holzapfel_disp_n_y.inp	Continuation of `xx_x1_holzapfel_disp.inp` with state imported.
xx_x1_holzapfel_load.inp	Base problem for carrying out import from Abaqus/Explicit to Abaqus/Explicit; uniaxial cyclic test; load control; C3D8R, CPE4R, CPS4R, M3D4R, and S4R elements.
xx_x2_holzapfel_load_n_y.inp	Continuation of `xx_x1_holzapfel_load.inp` with state imported.
xx_x1_holzapfel_visco.inp	Base problem for carrying out import from Abaqus/Explicit to Abaqus/Explicit; combination with viscoelastic and Mullins effect; uniaxial cyclic test; C3D8R, CPE4R, CPS4R, M3D4R, and S4R elements.
xx_x2_holzapfel_visco_n_y.inp	Continuation of `xx_x1_holzapfel_visco.inp` with state imported.

Anisotropic hyperelasticity with Holzapfel-Ogden strain energy function

Elements tested

C3D8R C3D10 C3D10HS CPE4R CPS4R M3D4R S4R

Problem description

Material:

Holzapfel-Ogden coefficients:

$a = 0.00099993$, $b = 3.143938$, $a_f = 0.004696677$, $b_f = 12.33052484$, $a_s = 0.002673816$, $b_s = 9.057685378$, $a_{fs} = 9.0234 \times 10^{-7}$, $b_{fs} = 0.000670602$.

Fiber directions (N=2):

$$A_1 = (\cos\gamma, \sin\gamma, 0),$$

$$A_2 = (-\sin\gamma, \cos\gamma, 0),$$

with γ=49.98°.

Compressible case: $D = 0.1$.

(The units are not important.)

Results and discussion

The results agree well with exact analytical or approximate solutions.

Input files

Abaqus/Standard input files

holzapfelogden_2fiber_std_uni.inp	Uniaxial tension; displacement control; C3D8, C3D10, and C3D20R elements.
holzapfelogden_2fiber_std_uni_c3d10hs.inp	Uniaxial tension; displacement control; C3D8, C3D10HS, and C3D20R elements.
holzapfelogden_2fiber_uni_hybrid.inp	Uniaxial tension; displacement control; C3D8H, C3D10H, and C3D20RH elements.
holzapfelogden_2fiber_std_uniori.inp	Uniaxial tension; displacement control; orientation along nonglobal Cartesian system; C3D8, C3D10, and C3D20R elements.
holzapfelogden_2fiber_std_uniori_c3d10hs.inp	Uniaxial tension; displacement control; orientation along nonglobal Cartesian system; C3D8, C3D10HS, and C3D20R elements.
holzapfelogden_c3d8_std_ss.inp	Simple shear; displacement control; C3D8 elements.
holzapfelogden_c3d8_std_uni.inp	Cyclic uniaxial tension; displacement control; C3D8 elements.
holzapfelogden_2fiber_pless.inp	Simple shear; displacement control; C3D8H, C3D8RH, and CPE4RH elements.
holzapfelogden_2fiber_ps.inp	Uniaxial tension; plane stress; load control; C3D8RH, CPS4R, M3D4R, and S4R elements.
sx_s_holzapfelogden.inp	Base problem for carrying out import from Abaqus/Standard to Abaqus/Explicit; C3D8R, CPE4R, CPS4R, M3D4, and S4R elements.
sx_x_holzapfelogden_n_y.inp	Continuation of `sx_s_holzapfelogden.inp` with state imported; C3D8R, CPE4R, CPS4R, M3D4, and S4R elements.

Abaqus/Explicit input files

holzapfelogden_disp_xpl.inp	Uniaxial cyclic test; displacement control; C3D8R, CPE4R, CPS4R, M3D4R, and S4R elements.
holzapfelogden_load_xpl.inp	Uniaxial cyclic test; load control; C3D8R, CPE4R, CPS4R, M3D4R, and S4R elements.
holzapfelogden_visco_xpl.inp	Combination with viscoelastic and Mullins effect; uniaxial cyclic test; C3D8R, CPE4R, CPS4R, M3D4R, and S4R elements.
xx_x1_holzapfelogden_disp.inp	Base problem for carrying out import from Abaqus/Explicit to Abaqus/Explicit; uniaxial cyclic test; displacement control; C3D8R, CPE4R, CPS4R, M3D4R, and S4R elements.
xx_x2_holzapfelogden_disp_n_y.inp	Continuation of `xx_x1_holzapfelogden_disp.inp` with state imported.
xx_x1_holzapfelogden_load.inp	Base problem for carrying out import from Abaqus/Explicit to Abaqus/Explicit; uniaxial cyclic test; load control; C3D8R, CPE4R, CPS4R, M3D4R, and S4R elements.

xx_x2_holzapfelogden_load_n_y.inp	Continuation of `xx_x1_holzapfelogden_load.inp` with state imported.
xx_x1_holzapfelogden_visco.inp	Base problem for carrying out import from Abaqus/Explicit to Abaqus/Explicit; combination with viscoelastic and Mullins effect; uniaxial cyclic test; C3D8R, CPE4R, CPS4R, M3D4R, and S4R elements.
xx_x2_holzapfelogden_visco_n_y.inp	Continuation of `xx_x1_holzapfelogden_visco.inp` with state imported.

Anisotropic hyperelasticity with Kaliske-Schmidt strain energy function

Elements tested

C3D8R C3D10 C3D10HS CPE4R CPS4R M3D4R S4R

Problem description

Material:

Kaliske-Schmidt coefficients:

$a_1 = 0.4 \times 10^9$, $b_1 = 0.2 \times 10^9$, $c_3 = 1.0 \times 10^9$, $d_2 = 0.3 \times 10^9$, $e_2 = 0.5 \times 10^9$, $f_3 = 1.0 \times 10^9$, $g_2 = 0.7 \times 10^8$.

Fiber directions (N=2):

$$A_1 = (\cos\gamma, \sin\gamma, 0),$$

$$A_2 = (\cos\gamma, -\sin\gamma, 0),$$

with γ=49.98°.

Compressible case: $D = 1 \times 10^{-11}$.

(The units are not important.)

Results and discussion

The results agree well with exact analytical or approximate solutions.

Input files

Abaqus/Standard input files

kaliske_2fiber_std_uni.inp	Uniaxial tension; displacement control; C3D8, C3D10, and C3D20R elements.
kaliske_2fiber_std_uni_c3d10hs.inp	Uniaxial tension; displacement control; C3D8, C3D10HS, and C3D20R elements.
kaliske_2fiber_uni_hybrid.inp	Uniaxial tension; displacement control; C3D8H, C3D10H, and C3D20RH elements.
kaliske_2fiber_std_uniori.inp	Uniaxial tension; displacement control; orientation along nonglobal Cartesian system; C3D8, C3D10, and C3D20R elements.
kaliske_2fiber_std_uniori_c3d10hs.inp	Uniaxial tension; displacement control; orientation along nonglobal Cartesian system; C3D8, C3D10HS, and C3D20R elements.
kaliske_c3d8_std_ss.inp	Simple shear; displacement control; C3D8 elements.
kaliske_c3d8_std_uni.inp	Cyclic uniaxial tension; displacement control; C3D8 elements.

kaliske_2fiber_ehgc.inp	Simple shear; displacement control; C3D8, C3D8R, and CPE4R elements.
kaliske_2fiber_pless.inp	Simple shear; displacement control; C3D8H, C3D8RH, and CPE4RH elements.
kaliske_2fiber_ps.inp	Uniaxial tension; plane stress; load control; C3D8RH, CPS4R, M3D4R, and S4R elements.
sx_s_kaliske.inp	Base problem for carrying out import from Abaqus/Standard to Abaqus/Explicit; C3D8R, CPE4R, CPS4R, M3D4, and S4R elements.
sx_x_kaliske_n_y.inp	Continuation of `sx_s_kaliske.inp` with state imported; C3D8R, CPE4R, CPS4R, M3D4, and S4R elements.

Abaqus/Explicit input files

kaliske_disp_xpl.inp	Uniaxial cyclic test; displacement control; C3D8R, CPE4R, CPS4R, M3D4R, and S4R elements.
kaliske_load_xpl.inp	Uniaxial cyclic test; load control; C3D8R, CPE4R, CPS4R, M3D4R, and S4R elements.
kaliske_visco_xpl.inp	Combination with viscoelastic and Mullins effect; uniaxial cyclic test; C3D8R, CPE4R, CPS4R, M3D4R, and S4R elements.
xx_x1_kaliske_disp.inp	Base problem for carrying out import from Abaqus/Explicit to Abaqus/Explicit; uniaxial cyclic test; displacement control; C3D8R, CPE4R, CPS4R, M3D4R, and S4R elements.
xx_x2_kaliske_disp_n_y.inp	Continuation of `xx_x1_kaliske_disp.inp` with state imported.
xx_x1_kaliske_visco.inp	Base problem for carrying out import from Abaqus/Explicit to Abaqus/Explicit; combination with viscoelastic and Mullins effect; uniaxial cyclic test; C3D8R, CPE4R, CPS4R, M3D4R, and S4R elements.
xx_x2_kaliske_visco_n_y.inp	Continuation of `xx_x1_kaliske_visco.inp` with state imported.

Constitutive Artificial Neural Networks (CANNs) models

Elements tested

C3D8 C3D8H C3D8R C3D8RH C3D10 C3D10H C3D20R C3D20RH
CPE4R CPS4R
M3D4R

Problem description

Input files implement the Neo-Hooke, Mooney-Rivlin, Holpzafel-Gasser-Ogden, Holzapfel-Ogden, and Kaliske-Schmidt hyperelastic models as constitutive artificial neural networks (CANNs) models.

Material:

Neo-Hookean coefficient: C_{10} = 100.

Compressible case: D_1 = 0.001.

(The units are not important.)

Mooney-Rivlin coefficients: C_{10} = 80., C_{01} = 20.

Compressible case: D_1 = 0.001.

(The units are not important.)

Holzapfel-Gasser-Ogden coefficients:

C_{10} = 7.64., k_1 = 996.6, k_2 = 524.6, κ = 0.226.

Fiber directions (N=2):

$$A_1 = (\cos\gamma, \sin\gamma, 0),$$

$$A_2 = (\cos\gamma, -\sin\gamma, 0),$$

with γ=49.98°.

Compressible case: $D = 1 \times 10^{-6}$.

(The units are not important.)

Holzapfel-Ogden coefficients:

a = 0.00099993, b = 3.143938, a_f = 0.004696677, b_f = 12.33052484, a_s = 0.002673816, b_s = 9.057685378, $a_{fs} = 9.0234 \times 10^{-7}$, b_{fs} = 0.000670602.

Fiber directions (N=2):

$$A_1 = (\cos\gamma, \sin\gamma, 0),$$

$$A_2 = (-\sin\gamma, \cos\gamma, 0),$$

with γ=49.98°.

Compressible case: D = 0.1.

(The units are not important.)

Kaliske-Schmidt coefficients:

$a_1 = 0.4 \times 10^9$, $b_1 = 0.2 \times 10^9$, $c_3 = 1.0 \times 10^9$, $d_2 = 0.3 \times 10^9$, $e_2 = 0.5 \times 10^9$, $f_3 = 1.0 \times 10^9$, $g_2 = 0.7 \times 10^8$.

Fiber directions (N=2):

$$A_1 = (\cos\gamma, \sin\gamma, 0),$$

$$A_2 = (\cos\gamma, -\sin\gamma, 0),$$

with γ=49.98°.

Compressible case: $D = 1 \times 10^{-11}$.

(The units are not important.)

Results and discussion

The results for the CANNs implementations agree with those for the native implementations.

Input files

Abaqus/Standard input files

mhncoo3hut_cann.inp	Incompressible Neo-Hooke model implemented as a CANNs model, uniaxial tension, C3D8RH element.

mhncdo3hut_cann.inp	Compressible Neo-Hooke model implemented as a CANNs model, uniaxial tension, C3D8R element.
mhmcoo3hut_cann.inp	Incompressible Mooney-Rivlin model implemented as a CANNs model, uniaxial tension, C3D8RH element.
mhmcdo3hut_cann.inp	Compressible Mooney-Rivlin model implemented as a CANNs model, uniaxial tension, C3D8R element.
hgo_2fiber_std_uni_cann.inp	Holzapfel-Gasser-Ogden model implemented as a CANNs model; uniaxial tension; displacement control; C3D8, C3D10, and C3D20R elements.
hgo_2fiber_uni_hybrid_cann.inp	Holzapfel-Gasser-Ogden model implemented as a CANNs model; uniaxial tension; displacement control; C3D8H, C3D10H, and C3D20RH elements.
hgo_2fiber_ps_cann.inp	Holzapfel-Gasser-Ogden model implemented as a CANNs model; uniaxial tension; plane stress; load control; C3D8RH, CPS4R, and M3D4R elements.
holzapfelogden_2fiber_std_uni_cann.inp	Holzapfel-Ogden model implemented as a CANNs model; uniaxial tension; displacement control; C3D8, C3D10, and C3D20R elements.
holzapfelogden_2fiber_uni_hybrid_cann.inp	Holzapfel-Ogden model implemented as a CANNs model; uniaxial tension; displacement control; C3D8H, C3D10H, and C3D20RH elements.
holzapfelogden_2fiber_ps_cann.inp	Holzapfel-Ogden model implemented as a CANNs model; uniaxial tension; plane stress; load control; C3D8RH, CPS4R, and M3D4R elements.
kaliske_2fiber_std_uni_cann.inp	Kaliske-Schmidt model implemented as a CANNs model; uniaxial tension; displacement control; C3D8, C3D10, and C3D20R elements.
kaliske_2fiber_uni_hybrid_cann.inp	Kaliske-Schmidt model implemented as a CANNs model; uniaxial tension; displacement control; C3D8H, C3D10H, and C3D20RH elements.
kaliske_2fiber_ps_cann.inp	Kaliske-Schmidt model implemented as a CANNs model; uniaxial tension; plane stress; load control; C3D8RH, CPS4R, and M3D4R elements.

Abaqus/Explicit input files

holzapfel_disp_cann_xpl.inp	Holzapfel-Gasser-Ogden model implemented as a CANNs model; uniaxial cyclic test; displacement control; C3D8R, CPE4R, CPS4R, and M3D4R elements.
holzapfelogden_disp_cann_xpl.inp	Holzapfel-Ogden model implemented as a CANNs model; uniaxial cyclic test; displacement control; C3D8R, CPE4R, CPS4R, and M3D4R elements.
kaliske_disp_cann_xpl.inp	Kaliske-Schmidt model implemented as a CANNs model; uniaxial cyclic test; displacement control; C3D8R, CPE4R, CPS4R, and M3D4R elements.

Input files containing definitions or data to be included in the input files listed above

ABQ_CANN_types.inp	Types of parameter tables used by the special-purpose built-in material for constitutive artificial neural networks (CANNs) models.

No compression

Elements tested

CPE4

Problem description

This option is used to modify the elasticity definition so that no compressive stress is allowed.

Material:

Young's modulus, $E = 3 \times 10^6$

Poisson's ratio, $\nu = 0.3$

Results and discussion

The results agree well with exact analytical or approximate solutions.

Input files

melnco1euc.inp **NO COMPRESSION*, CPE4 elements.

No tension

Elements tested
CPE4

Problem description

This option is used to modify the elasticity definition so that no tensile stress is allowed.

Material:

Young's modulus, $E = 3 \times 10^6$

Poisson's ratio, $\nu = 0.3$

Results and discussion

The results agree well with exact analytical or approximate solutions.

Input files

melnto1hut.inp **NO TENSION*, CPE4 elements.

Compression factor with traction elastic behavior

Elements tested
COH2D4 COH3D8

Problem description

This option is used to modify the elasticity definition for uncoupled traction-separation elastic behavior such that the stiffness in compression is a user-specified factor times the stiffness in tension. Two cohesive elements are tested in each model. One element is subjected to a tensile strain (separation), while the other element is subjected to a compressive strain (separation).

Material:

Stiffness in normal direction, $E_{nn} = 2 \times 10^5$

Compression factor = 2.0

Results and discussion

The results agree well with exact analytical solutions.

Input files

Abaqus/Standard input files

coh2d4_compfac.inp	COH2D4 elements.
coh3d8_compfac.inp	COH3D8 elements.

Abaqus/Explicit input files

coh2d4_compfac_xpl.inp	COH2D4 elements.
coh3d8_compfac_xpl.inp	COH3D8 elements.

Viscoelastic materials

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Large-strain time domain viscoelasticity with hyperelasticity

Elements tested

B31 CAX4R CPE4 CPE4H CPE4HT CPE4RH CPS4 CPS4R C3D8 C3D8RH C3D8RHT M3D4

Problem description

Material 1:

Polynomial coefficients (N=1): C_{10} = 8., C_{01} = 2.

Compressible case: D_1 = 0.1.

Prony series coefficients (N=1): $\overline{g}_1^P$ = 0., $\overline{k}_1^P$ = 0.5, τ_1 = 3.

Material 2:

Polynomial coefficients (N=1): C_{10} = 8., C_{01} = 2.

Compressible case: D_1 = 0.1.

Prony series coefficients (N=1): $\overline{g}_1^P$ = 0.5, $\overline{k}_1^P$ = 0., τ_1 = 3.

Heat transfer properties for coupled analysis: conductivity = 0.01, density = 1.

Specific heat = 1.

Material 3:

Polynomial coefficients (N=1): C_{10} = 1.5 × 10^6, C_{01} = 0.5 × 10^6.

Compressible case: D_1 = 1. × 10^{-7}.

Prony series coefficients (N=2):

$\overline{g}_1^P$ = 0.5, $\overline{k}_1^P$ = 0., τ_1 = 0.2.

$\overline{g}_2^P$ = 0.49, $\overline{k}_2^P$ = 0., τ_2 = 0.5.

Material 4:

Polynomial coefficients (N=1): C_{10} = 27.02, C_{01} = 1.42.

Compressible case: D_1 = 0.000001.

Prony series coefficients (N=2):

$\overline{g}_1^P$ = 0.25, $\overline{k}_1^P$ = 0.25, τ_1 = 5.

$\overline{g}_2^P$ = 0.25, $\overline{k}_2^P$ = 0.25, τ_2 = 10.

Creep compliance test data generated from Prony series above.

Stress relaxation test data generated from Prony series above.

Material 5:

Polynomial coefficients (N=1): C_{10} = 8., C_{01} = 2.

Compressible case: D_1 = 0.001.

Prony series coefficients (N=2):

$\overline{g}_1^P$ = 0.5, $\overline{k}_1^P$ = 0., τ_1 = 1.

$\overline{g}_2^P$ = 0.49, $\overline{k}_2^P$ = 0., τ_2 = 2.

Material 6:

Polynomial coefficients (N=1): C_{10} = 550.53, C_{01} = –275.265.

Compressible case: D_1 = $7. \times 10^{-7}$.

Prony series coefficients (N=6):

$\overline{g}_1^P$ = 0.1986, $\overline{k}_1^P$ = 0., τ_1 = 0.281×10^{-7}.

$\overline{g}_2^P$ = 0.1828, $\overline{k}_2^P$ = 0., τ_2 = 0.281×10^{-5}.

$\overline{g}_3^P$ = 0.1388, $\overline{k}_3^P$ = 0., τ_3 = 0.281×10^{-3}.

$\overline{g}_4^P$ = 0.2499, $\overline{k}_4^P$ = 0., τ_4 = 0.281×10^{-1}.

$\overline{g}_5^P$ = 0.1703, $\overline{k}_5^P$ = 0., τ_5 = 0.281×10^{1}.

$\overline{g}_6^P$ = 0.0593, $\overline{k}_6^P$ = 0., τ_6 = 0.281×10^{3}.

Material 7:

Ogden coefficients (N=2): μ_1 = 16., α_1 = 2., μ_2 = 4., α_2 = –2.

Prony series coefficients (N=1): $\overline{g}_1^P$ = 0.5, $\overline{k}_1^P$ = 0., τ_1 = 3.

Material 8:

Arruda-Boyce coefficients: $\mu = 20.$, $\lambda_m = 7.$

Prony series coefficients (N=1): $\overline{g}_1^P = 0.5$, $\overline{k}_1^P = 0.$, $\tau_1 = 3.$

Material 9:

Van der Waals coefficients: $\mu = 20.$, $\lambda_m = 10.$, $a = 0.1$, $\beta = 0.02$.

Prony series coefficients (N=1): $\overline{g}_1^P = 0.5$, $\overline{k}_1^P = 0.$, $\tau_1 = 3.$

Material 10:

Neo-Hookean coefficient: $c_{10} = 1$, $D = 0.1$.

Prony series coefficients (N=1): $\overline{g}_1^P = 0.5$, $\overline{k}_1^P = 0$, $\tau_1 = 0.1$.

Material 11:

Ogden coefficients (N=3): $\mu_1 = 64.26$, $\alpha_1 = 1.8$, $\mu_2 = 25.$, $\alpha_2 = -2.$, $\mu_3 = 18.76$, $\alpha_3 = 7$.

Prony series coefficients (N=1): $\overline{g}_1^P = 0.72$, $\overline{k}_1^P = 0.$, $\tau_1 = 17.5$.

Heat transfer properties for coupled analysis: conductivity = 1×10^{-6}, density = 7800, specific heat = 10, inelastic heat fraction = 0.8.

Results and discussion

The results agree well with exact analytical or approximate solutions.

Calibration of Prony series parameters from frequency-dependent moduli and vice versa has been tested for Materials 1, 4, and 6 in various relaxation and steady-state dynamic analyses. The data conversion is performed automatically in Abaqus. In the tests described below, some of the time domain analyses are repeated using frequency-dependent moduli data and some of the frequency domain (steady-state dynamic) analyses are repeated using time-dependent moduli data. The results of the repeated analyses are in good agreement with those of the original.

Input files

Material 1:

mvhcdo2ahc.inp	Compressible, volumetric compression, CPS4 elements.
mvhcdo2sr2.inp	Compressible, volumetric compression, CPS4 elements; Prony series parameters calibrated from frequency-dependent moduli.
mvhcdo2ssd.inp	Compressible, volumetric compression, CPS4 elements; steady-state dynamic, frequency-dependent moduli data derived from specified Prony series parameters.
mvhcdo2ss2.inp	Compressible, volumetric compression, CPS4 elements; steady-state dynamic, direct specification of frequency-dependent moduli data.

mvhcdo2zzz.inp Tabulated frequency-dependent moduli data included in mvhcdo2sr2.inp and mvhcdo2ss2.inp.

mvhcdo3ahc.inp Compressible, volumetric compression, CPE4 elements.

Material 2:

mvccoo3hut.inp Incompressible, uniaxial tension, coupled analysis, CPE4HT elements.

mvhcoo2rre.inp Incompressible, relaxation in uniaxial tension, CPS4 elements.

mvhcoo3hut.inp Incompressible, uniaxial tension, CPE4H elements.

mvhcoo3ltr.inp Incompressible, triaxial, CPE4H elements.

Material 3:

mvhcdo3rre.inp Compressible, relaxation in uniaxial tension, CPE4 elements.

Material 4:

mvhcdo3srs.inp Compressible, uniaxial tension, static and relaxation, CPE4H elements.

mvhtdo3srs.inp Creep and relaxation test data, uniaxial tension, static and relaxation, 2 CPE4RH elements.

mvhtdo3sr2.inp Compressible, uniaxial tension, static and relaxation, 2 CPE4RH elements; Prony series parameters calibrated from frequency-dependent moduli.

mvhtdo3ssd.inp Creep and relaxation test data, compressible, uniaxial tension, steady-state dynamic, 2 CPE4RH elements.

mvhtdo3ss2.inp Compressible, uniaxial tension, steady-state dynamic, 2 CPE4RH elements; direct specification of Prony series parameters calibrated in mvhtdo3ssd.inp.

mvhtdo3ss3.inp Compressible, uniaxial tension, steady-state dynamic, 2 CPE4RH elements; frequency-dependent moduli data derived from Prony series parameters calibrated from shear relaxation and creep test data as used in mvhtdo3ssd.inp.

mvhtdo3zzz.inp Tabulated frequency-dependent moduli data included in mvhtdo3ss3.inp.

mvhtdo3srs1.inp Combined test data, uniaxial tension, static and relaxation, 2 CPE4RH elements.

mvhcdo2srs.inp Compressible, uniaxial tension and rotation, static and relaxation, CPS4 elements.

mvhcdo2vlp.inp Compressible, uniaxial tension and rotation, static and relaxation with static linear perturbation steps containing **LOAD CASE*, CPS4R elements.

Material 5:

mvhcdo2rre.inp Compressible, relaxation in uniaxial tension, M3D4 elements.

Material 6:

mvhcdo3kct.inp Compressible, biaxial compression tension, CAX4R elements.

mvhcdo3kc2.inp Compressible, biaxial compression tension, CAX4R elements; Prony series parameters calibrated from frequency-dependent moduli.

mvhcdo3ssd.inp Compressible, biaxial compression tension, CAX4R elements; steady-state dynamic, frequency-dependent moduli data derived from specified Prony series parameters.

mvhcdo3ss2.inp Compressible, biaxial compression tension, CAX4R elements; steady-state dynamic, direct specification of frequency-dependent moduli data.

mvhcdo3zzz.inp Tabulated frequency-dependent moduli data used in mvhcdo3kc2.inp and mvhcdo3ss2.inp as an **INCLUDE* file.

Material 7:

mvhcoo3rre.inp Incompressible, relaxation in uniaxial tension, Ogden model, CPE4H elements.

mvhcoo3vlp.inp	Incompressible, uniaxial tension with static linear perturbation steps, Ogden model, CPE4H elements.
Material 8:	
mvacoo3rre.inp	Incompressible, relaxation in uniaxial tension, Arruda-Boyce model, CPE4H elements.
mvacoo3vlp.inp	Incompressible, uniaxial tension with static linear perturbation steps containing **LOAD CASE*, Arruda-Boyce model, CPE4H elements.
Material 9:	
mvvcoo3rre.inp	Incompressible, relaxation in uniaxial tension, Van der Waals model, CPE4H elements.
mvvcoo3vlp.inp	Incompressible, uniaxial tension with static linear perturbation steps, Van der Waals model, CPE4H elements.
Material 10:	
neoh_ve_unicyclic_b31.inp	Incompressible, uniaxial cyclic test with neo-Hookean model, B31 and C3D8RH elements.
neoh_ve_creep_b31.inp	Creep test with neo-Hookean model (compressible and incompressible), B31 and C3D8RH elements.
neoh_ve_relax_b31.inp	Relaxation test with neo-Hookean model (compressible and incompressible), B31 and C3D8RH elements.
Material 11:	
ogden_ve_ssh_cyclic.inp	Coupled temperature-displacement analysis with viscous dissipation as a heat source, incompressible, cyclic simple shear test, Ogden model, C3D8RHT element.

Large-strain time domain viscoelasticity with a hyperfoam material

Elements tested

CPE4

Problem description

Material 1:

Hyperfoam coefficients (N=3):

$\mu_1 = -17.4$, $\alpha_1 = -1.22$, $\mu_2 = 548.2$, $\alpha_2 = 17.3$, $\mu_3 = 10.47$, $\alpha_3 = -1.775$, $\nu_1 = \nu_2 = \nu_3 = 0$.

Prony series coefficients (N=1): $\overline{g}_1^P = 0.5$, $\overline{k}_1^P = 0.$, $\tau_1 = 3$.

Material 2:

Hyperfoam coefficients (N=3): Uniaxial test data, compression, Poisson's ratio = 0.

Prony series coefficients (N=1): $\overline{g}_1^P = 0.5$, $\overline{k}_1^P = 0.5$, $\tau_1 = 3$.

Material 3:

Hyperfoam coefficients (N=3):

$\mu_1 = -17.4$, $\alpha_1 = -1.22$, $\mu_2 = 548.2$, $\alpha_2 = 17.3$, $\mu_3 = 10.47$, $\alpha_3 = -1.775$, $\nu_1 = \nu_2 = \nu_3 = 0$.

Prony series coefficients (N=1): $\overline{g}_1^P = 0.5$, $\overline{k}_1^P = 0.$, $\tau_1 = 3$.

Mullins effect Ogden-Roxburgh model coefficients: $r = 2.0$, $m = 100.0$, $\beta = 0.5$.

Material 4:

Hyperfoam coefficients (N=3):

$\mu_1 = -17.4$, $\alpha_1 = -1.22$, $\mu_2 = 548.2$, $\alpha_2 = 17.3$, $\mu_3 = 10.47$, $\alpha_3 = -1.775$, $\nu_1 = \nu_2 = \nu_3 = 0$.

Prony series coefficients (N=1): $\overline{g}_1^P = 0.5$, $\overline{k}_1^P = 0.$, $\tau_1 = 3$.

Mullins effect slow decay model coefficients: $\eta_F = 0.5$, $U_h = 100.0$, $\alpha = 2.0$.

Results and discussion

The results agree well with exact analytical or approximate solutions.

Input files

Material 1 (Coefficient input):

mvfcdo3rre.inp	Compressible, relaxation in uniaxial tension, CPE4 elements.
mvfcdo3vlp.inp	Compressible, uniaxial tension with static linear perturbation steps containing **LOAD CASE*, CPE4 elements.

Material 2 (Test data input):

mvftdo3rre.inp	Compressible, relaxation in uniaxial tension, CPE4 elements.
mvftdo3rre_stbil_adap.inp	Compressible, relaxation in uniaxial tension, CPE4 elements; with adaptive stabilization.

Material 3:

hfoam_viscoela_mullins_cpe4.inp	Compressible, nonmonotonic uniaxial loading, CPE4 elements.
hfoam_viscoela_mullins_cps4.inp	Compressible, nonmonotonic uniaxial loading, CPS4 elements.
hfoam_viscoela_mullins_c3d8.inp	Compressible, nonmonotonic uniaxial loading, C3D8 elements.
hfoam_viscoela_mullins_c3d8_cycles.inp	Compressible, cyclic uniaxial loading, C3D8 elements.

Material 4:

hfoam_viscoela_mullins_slowdecay_cpe4.inp	Compressible, nonmonotonic uniaxial loading, CPE4 elements.
hfoam_viscoela_mullins_slowdecay_cps4.inp	Compressible, nonmonotonic uniaxial loading, CPS4 elements.
hfoam_viscoela_mullins_slowdecay_c3d8.inp	Compressible, nonmonotonic uniaxial loading, C3D8 elements.

Small-strain time domain viscoelasticity with linear elasticity

Elements tested

CPS4

Problem description

Material:

Young's modulus = 30.

Poisson's ratio = 0.4.

Prony series coefficients (N=2):

$\overline{g}_1^P$ = 0.25, $\overline{k}_1^P$ = 0.25, τ_1 = 5.

$\overline{g}_2^P$ = 0.25, $\overline{k}_2^P$ = 0.25, τ_2 = 10.

Creep compliance test data generated from Prony series above.

Stress relaxation test data generated from Prony series above.

Results and discussion

The results agree well with exact analytical or approximate solutions.

Input files

mvliso2srs.inp	Time domain viscoelasticity, elastic, CPS4 elements.

Small-strain time domain viscoelasticity with anisotropic elasticity

Elements tested

C3D8R CPE4R CPS4R S4R M3D4R

Problem description

The verification tests in this section consist of one-element relaxation tests with viscoelastic materials. The elements are loaded in tension or shear, followed by relaxation at constant strain.

Results and discussion

The results agree well with exact analytical or approximate solutions.

Input files

visco_ortho_relax.inp	Time-domain viscoelasticity with orthotropic elasticity.
visco_ortho_creep.inp	Time-domain viscoelasticity with orthotropic elasticity.
visco_aniso_relax.inp	Time-domain viscoelasticity with anisotropic elasticity.
visco_aniso_creep.inp	Time-domain viscoelasticity with anisotropic elasticity.

Small-strain time domain viscoelasticity with traction-separation elasticity

Elements tested

COH2D4 COH3D8

Problem description

This section includes verification tests for time domain viscoelasticity in combination with cohesive elements with traction-separation elasticity. One set of verification tests consists of relaxation tests in which the cohesive elements are loaded in the normal or shear directions, followed by relaxation at constant separation. Another set of verification tests is included for the combination of viscoelasticity with traction-separation elasticity and progressive damage. In these tests, the cohesive elements are loaded monotonically up to the point of failure.

Results and discussion

The results agree well with exact analytical or approximate solutions.

Input files

visco_coh2d.inp	Time domain viscoelasticity with traction-separation elasticity, COH2D4 elements.
visco_coh3d.inp	Time domain viscoelasticity with traction-separation elasticity, COH3D8 elements.
visco_dmg_coh2d.inp	Time domain viscoelasticity with traction-separation elasticity and damage, COH2D4 elements.
visco_dmg_coh3d.inp	Time domain viscoelasticity with traction-separation elasticity and damage, COH3D8 elements.

Frequency domain viscoelasticity

Elements tested

C3D8 CPS4

Problem description

Material 1:

Young's modulus = 200 GPa.

Poisson's ratio = 0.3.

Density = 8000 kg/m^3.

Fourier transform coefficients (tabular):

$\mathfrak{R}_1(wg) = 1.161 \times 10^{-2}$, $\mathfrak{I}_1(wg) = -3.21 \times 10^{-2}$, $\mathfrak{R}_1(wk) = 0$, $\mathfrak{I}_1(wk) = 0$, $f_1 = 1$.

$\mathfrak{R}_2(wg) = 7.849 \times 10^{-3}$, $\mathfrak{I}_2(wg) = -2.222 \times 10^{-2}$, $\mathfrak{R}_2(wk) = 0$, $\mathfrak{I}_2(wk) = 0$, $f_2 = 15.8$.

$\mathfrak{R}_3(wg) = 5.354 \times 10^{-3}$, $\mathfrak{I}_3(wg) = -1.533 \times 10^{-2}$, $\mathfrak{R}_3(wk) = 0$, $\mathfrak{I}_3(wk) = 0$, $f_3 = 25.1$.

$\mathfrak{R}_4(wg) = 3.639 \times 10^{-3}$, $\mathfrak{I}_4(wg) = -1.062 \times 10^{-2}$, $\mathfrak{R}_4(wk) = 0$, $\mathfrak{I}_4(wk) = 0$, $f_4 = 39.8$.

$\mathfrak{R}_5(wg) = 2.543 \times 10^{-3}$, $\mathfrak{I}_5(wg) = -7.382 \times 10^{-3}$, $\mathfrak{R}_5(wk) = 0$, $\mathfrak{I}_5(wk) = 0$, $f_5 = 63.1$.

$\mathfrak{R}_6(wg)$ = 1.775 × 10^{-3}, $\mathfrak{I}_6(wg)$ = –5.116 × 10^{-3}, $\mathfrak{R}_6(wk)$ = 0, $\mathfrak{I}_6(wk)$ = 0, f_6 = 100.

Material 2:

Young's modulus = 200 GPa.

Poisson's ratio = 0.3.

Density = 8000 kg/m^3.

Fourier transform coefficients (formula):

$\mathfrak{R}(g_1)$ = 2.3508 × 10^{-3}, $\mathfrak{I}(g_1)$ = 6.5001 × 10^{-3}, a = 1.38366, $\mathfrak{R}(k_1) = \mathfrak{I}(k_1) = b = 0$.

Material 3:

Polynomial coefficients (N=1): C_{10} = 33.333333 × 10^9, C_{01} = 0, D_1 = 12.0 × 10^{-12}.

Fourier transform coefficients (tabular):

$\mathfrak{R}_1(wg)$ = 1.161 × 10^{-2}, $\mathfrak{I}_1(wg)$ = –3.21 × 10^{-2}, $\mathfrak{R}_1(wk)$ = 0, $\mathfrak{I}_1(wk)$ = 0, f_1 = 1.

$\mathfrak{R}_2(wg)$ = 7.849 × 10^{-3}, $\mathfrak{I}_2(wg)$ = –2.222 × 10^{-2}, $\mathfrak{R}_2(wk)$ = 0, $\mathfrak{I}_2(wk)$ = 0, f_2 = 15.8.

$\mathfrak{R}_3(wg)$ = 5.354 × 10^{-3}, $\mathfrak{I}_3(wg)$ = –1.533 × 10^{-2}, $\mathfrak{R}_3(wk)$ = 0, $\mathfrak{I}_3(wk)$ = 0, f_3 = 25.1.

$\mathfrak{R}_4(wg)$ = 3.639 × 10^{-3}, $\mathfrak{I}_4(wg)$ = –1.062 × 10^{-2}, $\mathfrak{R}_4(wk)$ = 0, $\mathfrak{I}_4(wk)$ = 0, f_4 = 39.8.

$\mathfrak{R}_5(wg)$ = 2.543 × 10^{-3}, $\mathfrak{I}_5(wg)$ = –7.382 × 10^{-3}, $\mathfrak{R}_5(wk)$ = 0, $\mathfrak{I}_5(wk)$ = 0, f_5 = 63.1.

$\mathfrak{R}_6(wg)$ = 1.775 × 10^{-3}, $\mathfrak{I}_6(wg)$ = –5.116 × 10^{-3}, $\mathfrak{R}_6(wk)$ = 0, $\mathfrak{I}_6(wk)$ = 0, f_6 = 100.

Material 4:

Polynomial coefficients (N=1): C_{10} = 33.333333 × 10^9, C_{01} = 0, D_1 = 12.0 × 10^{-12}.

Fourier transform coefficients (formula):

$\mathfrak{R}(g_1)$ = 2.3508 × 10^{-3}, $\mathfrak{I}(g_1)$ = 6.5001 × 10^{-3}, a = 1.38366, $\mathfrak{R}(k_1) = \mathfrak{I}(k_1) = b = 0$.

Material 5:

Polynomial coefficients (N=1): C_{10} = 33.333333 × 10^9, C_{01} = 0, D_1 = 12.0 × 10^{-12}.

Fourier transform coefficients (formula):

$\mathfrak{R}(g_1)$ = 2.3508 × 10^{-3}, $\mathfrak{I}(g_1)$ = 6.5001 × 10^{-3}, a = 0, $\mathfrak{R}(k_1) = \mathfrak{I}(k_1) = b = 0$.

Material 6:

Arruda-Boyce coefficients: μ = 66.6666 × 10^9, λ_m = 5. , D = 12.0 × 10^{-12}.

Fourier transform coefficients (tabular):

$\mathfrak{R}_1(wg) = 1.161 \times 10^{-2}$, $\mathfrak{I}_1(wg) = -3.21 \times 10^{-2}$, $\mathfrak{R}_1(wk) = 0$, $\mathfrak{I}_1(wk) = 0$, $f_1 = 1.$

$\mathfrak{R}_2(wg) = 7.849 \times 10^{-3}$, $\mathfrak{I}_2(wg) = -2.222 \times 10^{-2}$, $\mathfrak{R}_2(wk) = 0$, $\mathfrak{I}_2(wk) = 0$, $f_2 = 15.8.$

$\mathfrak{R}_3(wg) = 5.354 \times 10^{-3}$, $\mathfrak{I}_3(wg) = -1.533 \times 10^{-2}$, $\mathfrak{R}_3(wk) = 0$, $\mathfrak{I}_3(wk) = 0$, $f_3 = 25.1.$

$\mathfrak{R}_4(wg) = 3.639 \times 10^{-3}$, $\mathfrak{I}_4(wg) = -1.062 \times 10^{-2}$, $\mathfrak{R}_4(wk) = 0$, $\mathfrak{I}_4(wk) = 0$, $f_4 = 39.8.$

$\mathfrak{R}_5(wg) = 2.543 \times 10^{-3}$, $\mathfrak{I}_5(wg) = -7.382 \times 10^{-3}$, $\mathfrak{R}_5(wk) = 0$, $\mathfrak{I}_5(wk) = 0$, $f_5 = 63.1.$

$\mathfrak{R}_6(wg) = 1.775 \times 10^{-3}$, $\mathfrak{I}_6(wg) = -5.116 \times 10^{-3}$, $\mathfrak{R}_6(wk) = 0$, $\mathfrak{I}_6(wk) = 0$, $f_6 = 100.$

Material 7:

Arruda-Boyce coefficients: $\mu = 66.6666 \times 10^9$, $\lambda_m = 5.$, $D = 12.0 \times 10^{-12}$.

Fourier transform coefficients (formula):

$\mathfrak{R}(g_1) = 2.3508 \times 10^{-3}$, $\mathfrak{I}(g_1) = 6.5001 \times 10^{-3}$, $a = 1.38366$, $\mathfrak{R}(k_1) = \mathfrak{I}(k_1) = b = 0.$

Material 8:

Van der Waals coefficients: $\mu = 66.6666 \times 10^9$, $\lambda_m = 10.$, $a = 0.1$, $\beta = 0.$, $D = 12.0 \times 10^{-12}$.

Fourier transform coefficients (tabular):

$\mathfrak{R}_1(wg) = 1.161 \times 10^{-2}$, $\mathfrak{I}_1(wg) = -3.21 \times 10^{-2}$, $\mathfrak{R}_1(wk) = 0$, $\mathfrak{I}_1(wk) = 0$, $f_1 = 1.$

$\mathfrak{R}_2(wg) = 7.849 \times 10^{-3}$, $\mathfrak{I}_2(wg) = -2.222 \times 10^{-2}$, $\mathfrak{R}_2(wk) = 0$, $\mathfrak{I}_2(wk) = 0$, $f_2 = 15.8.$

$\mathfrak{R}_3(wg) = 5.354 \times 10^{-3}$, $\mathfrak{I}_3(wg) = -1.533 \times 10^{-2}$, $\mathfrak{R}_3(wk) = 0$, $\mathfrak{I}_3(wk) = 0$, $f_3 = 25.1.$

$\mathfrak{R}_4(wg) = 3.639 \times 10^{-3}$, $\mathfrak{I}_4(wg) = -1.062 \times 10^{-2}$, $\mathfrak{R}_4(wk) = 0$, $\mathfrak{I}_4(wk) = 0$, $f_4 = 39.8.$

$\mathfrak{R}_5(wg) = 2.543 \times 10^{-3}$, $\mathfrak{I}_5(wg) = -7.382 \times 10^{-3}$, $\mathfrak{R}_5(wk) = 0$, $\mathfrak{I}_5(wk) = 0$, $f_5 = 63.1.$

$\mathfrak{R}_6(wg) = 1.775 \times 10^{-3}$, $\mathfrak{I}_6(wg) = -5.116 \times 10^{-3}$, $\mathfrak{R}_6(wk) = 0$, $\mathfrak{I}_6(wk) = 0$, $f_6 = 100.$

Material 9:

Van der Waals coefficients: $\mu = 66.6666 \times 10^9$, $\lambda_m = 10.$, $a = 0.1$, $\beta = 0.$, $D = 12.0 \times 10^{-12}$.

Fourier transform coefficients (formula):

$\mathfrak{R}(g_1) = 2.3508 \times 10^{-3}$, $\mathfrak{I}(g_1) = 6.5001 \times 10^{-3}$, $a = 1.38366$, $\mathfrak{R}(k_1) = \mathfrak{I}(k_1) = b = 0.$

Material 10:

Orthotropic elastic material coefficients:

D_{1111}=269.5×10^9, D_{1122}=132.0×10^9, D_{2222}=319.0×10^9

$D_{1133}=99.5\times10^9$, $D_{2233}=50.0\times10^9$, $D_{3333}=351.5\times10^9$

$D_{1212}=80.0\times10^9$, $D_{1313}=90.0\times10^9$, $D_{2323}=110.0\times10^9$

Density = 8000 kg/m^3.

Fourier transform coefficients (formula):

$\Re(g_1)$ = 2.3508 × 10^{-3}, $\Im(g_1)$ = 6.5001 × 10^{-3}, a = 1.38366, $\Re(k_1)=\Im(k_1)=b=0$.

Material 11:

Anisotropic elastic material nonzero coefficients:

$D_{1111}=269.5\times10^9$, $D_{1122}=132.0\times10^9$, $D_{2222}=319.0\times10^9$

$D_{1133}=99.5\times10^9$, $D_{2233}=50.0\times10^9$, $D_{3333}=351.5\times10^9$

$D_{1212}=80.0\times10^9$, $D_{1313}=90.0\times10^9$, $D_{2323}=110.0\times10^9$

Density = 8000 kg/m^3.

Fourier transform coefficients (tabular):

$\Re_1(wg)$ = 1.161 × 10^{-2}, $\Im_1(wg)$ = −3.21 × 10^{-2}, $\Re_1(wk)=0$, $\Im_1(wk)=0$, $f_1=1$.

$\Re_2(wg)$ = 7.849 × 10^{-3}, $\Im_2(wg)$ = −2.222 × 10^{-2}, $\Re_2(wk)=0$, $\Im_2(wk)=0$, $f_2=15.8$.

$\Re_3(wg)$ = 5.354 × 10^{-3}, $\Im_3(wg)$ = −1.533 × 10^{-2}, $\Re_3(wk)=0$, $\Im_3(wk)=0$, $f_3=25.1$.

$\Re_4(wg)$ = 3.639 × 10^{-3}, $\Im_4(wg)$ = −1.062 × 10^{-2}, $\Re_4(wk)=0$, $\Im_4(wk)=0$, $f_4=39.8$.

$\Re_5(wg)$ = 2.543 × 10^{-3}, $\Im_5(wg)$ = −7.382 × 10^{-3}, $\Re_5(wk)=0$, $\Im_5(wk)=0$, $f_5=63.1$.

$\Re_6(wg)$ = 1.775 × 10^{-3}, $\Im_6(wg)$ = −5.116 × 10^{-3}, $\Re_6(wk)=0$, $\Im_6(wk)=0$, $f_6=100$.

Results and discussion

The problem involves a direct-integration steady-state dynamic procedure in which a harmonic pressure of amplitude 1.0 GPa is applied to the top surface of a cantilevered beam. Several subspace-based steady-state dynamic procedures follow. The results of most interest are the vertical displacement at the tip of the cantilever and the phase angles of the displacements for the specified frequencies.

Input files

Material 1:

mveft02the.inp	Tabular frequency domain viscoelasticity, elastic, CPS4 elements.
mveft03the.inp	Tabular frequency domain viscoelasticity, elastic, C3D8 elements.

Material 2:

mveff02the.inp	Formula frequency domain viscoelasticity, elastic, CPS4 elements.
mveff03the.inp	Formula frequency domain viscoelasticity, elastic, C3D8 elements.

Material 3:

mvyft02the.inp	Tabular frequency domain viscoelasticity, hyperelastic, CPS4 elements.

mvyft03the.inp	Tabular frequency domain viscoelasticity, hyperelastic, C3D8 elements.
Material 4:	
mvyff02the.inp	Formula frequency domain viscoelasticity, hyperelastic, CPS4 elements.
mvyff03the.inp	Formula frequency domain viscoelasticity, hyperelastic, C3D8 elements.
Material 5:	
mvyfn02the.inp	Formula frequency domain viscoelasticity, hyperelastic, CPS4 elements.
mvyfn03the.inp	Formula frequency domain viscoelasticity, hyperelastic, C3D8 elements.
Material 6:	
mvxft02the.inp	Tabular frequency domain viscoelasticity, hyperelastic, CPS4 elements.
mvxft03the.inp	Tabular frequency domain viscoelasticity, hyperelastic, C3D8 elements.
Material 7:	
mvxfn02the.inp	Formula frequency domain viscoelasticity, hyperelastic, CPS4 elements.
mvxfn03the.inp	Formula frequency domain viscoelasticity, hyperelastic, C3D8 elements.
Material 8:	
mvzft02the.inp	Tabular frequency domain viscoelasticity, hyperelastic, CPS4 elements.
mvzft03the.inp	Tabular frequency domain viscoelasticity, hyperelastic, C3D8 elements.
Material 9:	
mvzfn02the.inp	Formula frequency domain viscoelasticity, hyperelastic, CPS4 elements.
mvzfn03the.inp	Formula frequency domain viscoelasticity, hyperelastic, C3D8 elements.
Material 10:	
frq_visco_formula_ortho_c3d8.inp	Formula frequency domain viscoelasticity, orthotropic elastic material, C3D8 elements.
frq_visco_formula_ortho_cpe4.inp	Formula frequency domain viscoelasticity, orthotropic elastic material, CPE4 elements.
Material 11:	
frq_visco_tabular_aniso_c3d8.inp	Tabular frequency domain viscoelasticity, anisotropic elastic material, C3D8 elements.
frq_visco_tabular_aniso_cpe4.inp	Tabular frequency domain viscoelasticity, anisotropic elastic material, CPE4 elements.

Frequency domain viscoelasticity defined directly in terms of storage and loss moduli

Elements tested

C3D8R C3D8RH

Problem description

In addition to the approach adopted in the verification problems of the earlier subsection, Abaqus allows definition of viscoelastic behavior in the frequency domain directly in terms of storage and loss moduli (as opposed to defining the viscoelastic behavior in terms of ratios that involve the long-term elastic shear and bulk moduli). The viscoelastic behavior can be defined using storage and loss moduli data obtained directly from a uniaxial tension test. Volumetric relaxation, if important, can also be defined in terms of bulk storage and loss moduli,

obtained directly from a volumetric test. In both cases, the viscoelastic properties can be defined in tabular forms as functions of frequency and level of preload. The problems described in this subsection use this approach.

The basic test setup consists of a reference element and a test element. For the reference element, the viscoelastic behavior is defined using the approach used in the previous subsection (that is, in terms of ratios that involve the long-term elastic modulus). For the test element, the viscoelastic behavior is defined directly in terms of uniaxial storage and loss moduli (and in some cases, bulk storage and loss moduli). However, in the latter case, the values of the uniaxial (and bulk) storage/loss moduli are hand-calculated based on the ratios specified for the reference element and the (preload-dependent) long-term elastic modulus. In computing the storage and loss moduli for the test case, it is assumed that the ratios specified for the reference case are independent of the level of preload. Since the purpose of the problems in this section is simply to verify that the implementation is correct, the aforementioned assumption should not be viewed as a limitation. Both the reference elements and the test elements are subjected to displacement-based harmonic excitations about an unloaded state as well as several levels of uniaxial and volumetric prestrain. The steady-state dynamic response is obtained in each case.

Results and discussion

By design, the reference elements and the test elements are expected to result in identical real and imaginary stresses. This acts as a verification for the implementation of the current approach.

Input files

frq_visco_prldu_ab.inp	Only uniaxial viscoelastic data specified, long-term elastic behavior defined using the Arruda-Boyce hyperelasticity model.
frq_visco_prldu_marlow.inp	Only uniaxial viscoelastic data specified, long-term elastic behavior defined using the Marlow hyperelasticity model.
frq_visco_prldu_poly1.inp	Only uniaxial viscoelastic data specified, long-term elastic behavior defined using the Mooney-Rivlin hyperelasticity model.
frq_visco_prldu_ogden.inp	Only uniaxial viscoelastic data specified, long-term elastic behavior defined using the third-order Ogden hyperelasticity model.
frq_visco_prldu_poly3.inp	Only uniaxial viscoelastic data specified, long-term elastic behavior defined using the third-order polynomial hyperelasticity model.
frq_visco_prldu_vdw.inp	Only uniaxial viscoelastic data specified, long-term elastic behavior defined using the Van der Waals hyperelasticity model.
frq_visco_prldu_hfoam.inp	Only uniaxial viscoelastic data specified, long-term elastic behavior defined using the second-order hyperfoam model.
frq_visco_prlduv_poly1.inp	Both uniaxial and volumetric viscoelastic data specified, long-term elastic behavior defined using the Mooney-Rivlin hyperelasticity model.
frq_visco_prlduv_poly3.inp	Both uniaxial and volumetric viscoelastic data specified, long-term elastic behavior defined using the third-order polynomial hyperelasticity model.
frq_visco_prlduv_hfoam.inp	Both uniaxial and volumetric viscoelastic data specified, long-term elastic behavior defined using the second-order hyperfoam model.
frq_visco_interp.inp	A basic test for interpolation of material properties.

Long-term stress initial conditions

Elements tested

C3D8R C3D8RH S4R T3D2

Problem description

This set of problems tests the correctness of prescribing long-term stress initial conditions for viscoelastic materials. All the tests use the same viscoelastic material with small relaxation times for each term in the Prony series that causes the stress to relax immediately. For each combination of a material, an element, and a procedure, three input files are used. The base input file is a single element model, which is subjected to uniaxial nonmonotonic loading. The other input files use the initial conditions capability to read long-term initial stresses and, if Mullins effect model is specified, the initial maximum energy density is from the base input file. In these files the initial conditions are specified using result values from a point on the loading or unloading path in the first analysis. The second analysis continues from this point with the same loading as the base analysis, and the results produced by both analyses are compared and are expected to be the same.

Material 1:

Hencky coefficients: E=160, ν= 0.5
Exponential model coefficients: η_F=0.4, U_h=10.0, α=1.0
Prony series coefficients (N=2): $\overline{g}_1^P$ = 0.31, $\overline{k}_1^P$ = 0.2, τ_1=2.0× 10^{-6}, $\overline{g}_2^P$ = 0.19, $\overline{k}_2^P$ = 0.1, τ_2=1.0× 10^{-5}

Material 2:

Hencky coefficients: E=160, ν= 0.5
Slow decay model coefficients: η_F=0.4, U_h=10.0, α=1.0
Prony series coefficients (N=2): $\overline{g}_1^P$ = 0.31, $\overline{k}_1^P$ = 0.2, τ_1=2.0× 10^{-6}, $\overline{g}_2^P$ = 0.19, $\overline{k}_2^P$ = 0.1, τ_2=1.0× 10^{-5}

Material 3:

Hyperfoam coefficients (N=1): μ_1=160.0, α_1=2.0, ν_1= 0.3
Prony series coefficients (N=2): $\overline{g}_1^P$ = 0.31, $\overline{k}_1^P$ = 0.2, τ_1=2.0× 10^{-6}, $\overline{g}_2^P$ = 0.19, $\overline{k}_2^P$ = 0.1, τ_2=1.0× 10^{-5}

Results and discussion

The results from the base analyses agree well with the results from the second analyses with initial conditions.

Input files

hencky_ve_exp_cycle_lt_t3d2.inp	Standard analysis, incompressible Hencky model, exponential model, viscoelasticity, T3D2 element.
hencky_ve_exp_cycle_lt_t3d2_load_ic.inp	Standard analysis, incompressible Hencky model, exponential model, viscoelasticity, T3D2 element, initial conditions specified using result values from a point on the loading path in the first analysis.
hencky_ve_exp_cycle_lt_t3d2_unload_ic.inp	Standard analysis, incompressible Hencky model, exponential model, viscoelasticity, T3D2 element, initial conditions specified using result values from a point on the unloading path in the first analysis.
hencky_ve_slowdecay_cycle_lt_c3d8rh.inp	Standard analysis, incompressible Hencky model, slow decay model, viscoelasticity, C3D8RH element.

hencky_ve_slowdecay_cycle_lt_c3d8rh_load_ic.inp	Standard analysis, incompressible Hencky model, slow decay model, viscoelasticity, C3D8RH element, initial conditions specified using result values from a point on the loading path in the first analysis.
hencky_ve_slowdecay_cycle_lt_c3d8rh_unload_ic.inp	Standard analysis, incompressible Hencky model, slow decay model, viscoelasticity, C3D8RH element, initial conditions specified using result values from a point on the unloading path in the first analysis.
hfoam_ve_cycle_lt_c3d8r.inp	Standard analysis, hyperfoam model, viscoelasticity, C3D8R element.
hfoam_ve_cycle_lt_c3d8r_load_ic.inp	Standard analysis, hyperfoam model, viscoelasticity, C3D8R element, initial conditions specified using result values from a point on the loading path in the first analysis.
hfoam_ve_cycle_lt_c3d8r_unload_ic.inp	Standard analysis, hyperfoam model, viscoelasticity, C3D8R element, initial conditions specified using result values from a point on the unloading path in the first analysis.
hfoam_ve_cycle_lt_s4r.inp	Standard analysis, hyperfoam model, viscoelasticity, S4R element.
hfoam_ve_cycle_lt_s4r_load_ic.inp	Standard analysis, hyperfoam model, viscoelasticity, S4R element, initial conditions specified using result values from a point on the loading path in the first analysis.
hfoam_ve_cycle_lt_s4r_unload_ic.inp	Standard analysis, hyperfoam model, viscoelasticity, S4R element, initial conditions specified using result values from a point on the unloading path in the first analysis.
hfoam_ve_cycle_lt_t3d2.inp	Standard analysis, hyperfoam model, viscoelasticity, T3D2 element.
hfoam_ve_cycle_lt_t3d2_load_ic.inp	Standard analysis, hyperfoam model, viscoelasticity, T3D2 element, initial conditions specified using result values from a point on the loading path in the first analysis.
hfoam_ve_cycle_lt_t3d2_unload_ic.inp	Standard analysis, hyperfoam model, viscoelasticity, T3D2 element, initial conditions specified using result values from a point on the unloading path in the first analysis.
x_hencky_ve_slowdecay_cycle_lt_c3d8r.inp	Explicit dynamic analysis, compressible Hencky model, slow decay model, C3D8R element.
x_hencky_ve_slowdecay_cycle_lt_c3d8r_load_ic.inp	Explicit dynamic analysis, compressible Hencky model, slow decay model, C3D8R element, initial conditions specified using result values from a point on the loading path in the first analysis.
x_hencky_ve_slowdecay_cycle_lt_c3d8r_unload_ic.inp	Explicit dynamic analysis, compressible Hencky model, slow decay model, C3D8R element, initial conditions specified using result values from a point on the unloading path in the first analysis.
x_hfoam_ve_cycle_lt_c3d8r.inp	Explicit dynamic analysis, hyperfoam model, viscoelasticity, C3D8R element.
x_hfoam_ve_cycle_lt_c3d8r_load_ic.inp	Explicit dynamic analysis, hyperfoam model, viscoelasticity, C3D8R element, initial conditions specified using result values from a point on the loading path in the first analysis.
x_hfoam_ve_cycle_lt_c3d8r_unload_ic.inp	Explicit dynamic analysis, hyperfoam model, viscoelasticity, C3D8R, initial conditions specified using result values from a point on the unloading path in the first analysis.

Mullins effect and permanent set

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Mullins effect in elastomers

Elements tested

SAX1 CPS4R CPE4R CPE4RH C3D8R C3D8RH T2D2

Problem description

The problems in this set can be broadly classified under three categories. The first category of problems consists of simple displacement- or load-controlled cyclic tests to verify the Mullins effect, with the primary response defined by different strain energy potential functions. The tests consist of a single element that is cyclically loaded to a maximum strain (stress) level, then unloaded to zero strain (stress). This is followed by further reloading to levels of strain (stress) that are higher than those reached during the loading segment of the first cycle, followed again by unloading to zero strain (stress). The tests in this section use parts and assemblies.

The second category of problems is intended for testing the calibration capabilities for determining the Mullins effect coefficients. The problems use unloading test data that were generated by running a model with specified values of the Mullins effect coefficients. The calibration capability is meant to recover the specified values of the Mullins effect coefficients. These tests use different loading states, such as uniaxial tension, biaxial tension, and planar tension.

The third category of problems tests the import capability with the Mullins effect. All tests in this section are set up with a uniaxial stress state. The tests consist of first loading a single element in Abaqus/Standard and unloading it. The results are then imported into Abaqus/Explicit, where the element is loaded to deformation levels higher than the original loading and then unloaded. These results are again imported back into Abaqus/Standard, where the element is loaded to deformation levels higher than the prior loading and then unloaded. In addition for the Ogden-Roxburgh model, the last set of results are imported from Abaqus/Standard to Abaqus/Standard, and the element is further deformed and unloaded. The above series of tests includes problems that import both the state and the reference configuration, problems that import only the state, and problems that import neither the state nor the reference configuration.

Material: The following material data are used for the first category of tests:

Strain energy potential form	Primary hyperelastic coefficients	Ogden-Roxburgh model parameters	Exponential and slow decay model parameters
Compressible Arruda-Boyce	μ = 200.0, λ_m = 5.0, D_1 = 0.001	r = 1.1, m = 100.0, β = 0.1	η_F = 0.5, U_h = 100.0, α = 1.5
Compressible Ogden	μ_1 = 160.0, α_1 = 2.0, μ_2 = 40.0, α_2 = –2.0, D_1 = 0.001	r = 5.0, m = 220.0, β = 0.1	N/A
Incompressible Ogden	μ_1 = 160.0, α_1 = 2.0, μ_2 = 40.0, α_2 = –2.0	r = 5.0, m = 220.0	η_F = 0.8, U_h = 220.0, α= 2.0
User-defined hyperelastic material	Same as the compressible Yeoh model		

Strain energy potential form	Primary hyperelastic coefficients	Ogden-Roxburgh model parameters	Exponential and slow decay model parameters
Compressible Van der Waals	μ = 200.0, λ_m = 10.0, α = 0.1, β = 0.0, D = 0.001	r = 3.0, m = 100.0, β = 0.1	η_F = 0.7, U_h = 100.0, α = 0.5
Compressible Yeoh	C_{10} = 1.326, C_{20} = –0.326, C_{30} = 0.1319, D_1 = 0.000725	r = 1.1, m = 100.0, β = 0.1	η_F = 0.5, U_h = 100.0, α = 1.5
Incompressible Yeoh	C_{10} = 1.326, C_{20} = –0.326, C_{30} = 0.1319	r = 1.1, m = 100.0	N/A

For the second and third category of tests the primary material response is defined using the incompressible Yeoh potential with the deviatoric coefficients as given above. For the second category of tests the unloading test data are generated for uniaxial, biaxial, and planar stress states using the following values for the Mullins effect parameters: r = 1.25, m = 0.01, and β = 0.9. These parameters are also used to define the Mullins effect in the third category of tests.

Loading: The first category of problems includes both displacement- and force-controlled loading. The second and third categories of problems are carried out under only displacement-controlled loading.

Results and discussion

For the first category of problems the results of the Abaqus/Standard and Abaqus/Explicit numerical simulations are in good agreement with the analytical results.

For the second category of problems, which tests the calibration of the Mullins effect parameters, it is observed that the parameters r and β are always captured accurately. A good fit for m is obtained in situations where the deformation level leads to a relatively large value of maximum deviatoric strain energy density, U^m_{dev}, such that the value of βU^m_{dev} dominates over the value of m.

For the final category of problems, which tests the import capability, the response after each import of results is as expected. When the state is imported, further deformation upon import shows the appropriate level of stress softening. On the other hand, when the state is not imported, no stress softening is observed.

Input files

Ogden-Roxburgh model

mmecdo2cut_arruda.inp	Compressible Arruda-Boyce model, CPE4RH element, cyclic uniaxial tension.
mmecdo2cut_vdw.inp	Compressible Van der Waals model, CPE4RH element, cyclic uniaxial tension.
mmecdo2cut_yeoh.inp	Compressible Yeoh model, SAX1 element, cyclic uniaxial tension.
mmecdo2cut.inp	Compressible Yeoh model, CPS4R element, cyclic uniaxial tension, tests temperature- and field-variable-dependent Mullins effect material properties.
mmecdo2cut_po.inp	Tests the **POST OUTPUT* capability for the damage-related output variables. This job needs the restart file from the job mmecdo2cut.inp.
mmecoo2cut_yeoh.inp	Incompressible Yeoh model, SAX1 element, cyclic uniaxial tension.
mmecoo2cut_user.inp	Incompressible Yeoh model, CPS4R element, cyclic uniaxial tension. The Mullins effect is implemented with user subroutine *UMULLINS*; the use of solution- dependent state variables in *UMULLINS* is also tested (the

	solution-dependent state variables are used to provide a nonzero initial value of U^m_{dev}).
mmecoo2cut_user.f	User subroutine *UMULLINS* to define damage variable for the Mullins effect material model.
mmecdo3cut_ogden.inp	Compressible Ogden model, C3D8RH element, cyclic uniaxial tension.
mmecoo3cut_ogden.inp	Incompressible Ogden model, C3D8RH element, cyclic uniaxial tension.
mmecdo3cut_user.inp	Compressible user-defined hyperelastic material, C3D8RH element, cyclic uniaxial tension, user subroutine *UHYPER* provided in the file mmecdo3cut_user.f.
mmecdo3cut_user.f	User subroutine *UHYPER* to define a compressible hyperelastic material model.
mmecdo3cut_yeoh.inp	Compressible Yeoh model, C3D8RH element, cyclic uniaxial tension.
mmecdo3cut_yeoh_load.inp	Compressible Yeoh model, C3D8RH element, cyclic uniaxial tension with load control.
mmecdo3ctu.inp	Compressible Yeoh model, C3D8RH element, triaxial tension followed by unloading, further loading in uniaxial tension, and unloading. The purpose of this test is to demonstrate that a purely volumetric deformation does not result in any damage. This problem also tests a static linear perturbation analysis about a damaged base state.
mmecoo3cut_yeoh.inp	Incompressible Yeoh model, C3D8RH element, cyclic uniaxial tension.
mmecdo2cut_arruda_visco.inp	Compressible Arruda-Boyce model with viscoelasticity, CPE4RH element, cyclic uniaxial tension.
mmecdo3cut_ogden_visco.inp	Compressible Ogden model with viscoelasticity, C3D8RH element, cyclic uniaxial tension.
mmecoo2cut_yeoh_visco.inp	Incompressible Yeoh model with viscoelasticity, SAX1 element, cyclic uniaxial tension.
mmecoo3cut_ogden_visco.inp	Incompressible Ogden model with viscoelasticity, C3D8RH element, cyclic uniaxial tension.
neoh_mullins_ve.inp	Neo-Hookean model with viscoelasticity and Mullins effect; C3D8R, CPE4R, and CPS4R elements; uniaxial loading-unloading at different strain levels.
x_mmecdo2cut_arruda.inp	Explicit dynamic test with compressible Arruda-Boyce model, CPE4RH element, cyclic uniaxial tension.
x_mmecdo2cut_vdw.inp	Explicit dynamic test with compressible Van der Waals model, CPE4RH element, cyclic uniaxial tension.
x_mmecdo3cut_ogden.inp	Explicit dynamic test with compressible Ogden model, C3D8RH element, cyclic uniaxial tension.
x_mmecdo2cut_yeoh.inp	Explicit dynamic test with compressible Yeoh model, SAX1 element, cyclic uniaxial tension.
x_mmecdo3cut_yeoh.inp	Explicit dynamic test with compressible Yeoh model, C3D8RH element, cyclic uniaxial tension.
x_mmecdo2cut_visarruda.inp	Explicit dynamic test with compressible Arruda-Boyce model and viscoelasticity, CPE4RH element, cyclic uniaxial tension.
x_mmecdo2cut_visvdw.inp	Explicit dynamic test with compressible Van der Waals model and viscoelasticity, CPE4RH element, cyclic uniaxial tension.
x_mmecdo3cut_visogden.inp	Explicit dynamic test with compressible Ogden model and viscoelasticity, C3D8RH element, cyclic uniaxial tension.

x_mmecdo3cut_visyeoh.inp	Explicit dynamic test with compressible Yeoh model and viscoelasticity, C3D8RH element, cyclic uniaxial tension.
x_mmecdo3cut_user.inp	Explicit dynamic test with compressible Yeoh model and viscoelasticity, C3D8RH element, cyclic uniaxial tension. The Mullins effect is implemented with user subroutine *VUMULLINS*.
x_mmecdo3cut_user.f	User subroutine *VUMULLINS* to define damage variable for the Mullins effect material model.
mmetdo3cut.inp	Calibration test with uniaxial unloading test data, C3D8RH element, cyclic uniaxial tension.
mmetdo3cut_marlow.inp	Calibration test with uniaxial unloading test data, C3D8RH element, cyclic uniaxial tension, Marlow model.
mmetdo3cbt.inp	Calibration test with biaxial unloading test data, C3D8RH element, cyclic biaxial tension.
mmetdo3cpt.inp	Calibration test with planar unloading test data, C3D8RH element, cyclic planar tension.
mmetdo3cpt_mult.inp	Calibration test with unloading test data from uniaxial, biaxial, and planar tests; C3D8RH element; cyclic planar tension.
mmetdo3cpt_r.inp	Calibration test with unloading test data from uniaxial, biaxial, and planar tests and with the value of the parameter r fixed; C3D8RH element; cyclic planar tension.
mmetdo3cpt_m.inp	Calibration test with unloading test data from uniaxial, biaxial, and planar tests and with the value of the parameter m fixed; C3D8RH element; cyclic planar tension.
mmetdo3cpt_beta.inp	Calibration test with unloading test data from uniaxial, biaxial, and planar tests and with the value of the parameter β fixed; C3D8RH element; cyclic planar tension.
sx_s_mullins.inp	Base problem for carrying out import from Abaqus/Standard to Abaqus/Explicit, C3D8RH element, cyclic uniaxial tension.
sx_x_mullins_y_y.inp	Explicit dynamic continuation of sx_s_mullins.inp with both the reference configuration and the state imported, C3D8RH element, cyclic uniaxial tension.
sx_x_mullins_n_y.inp	Explicit dynamic continuation of sx_s_mullins.inp with only the state imported, C3D8RH element, cyclic uniaxial tension.
sx_x_mullins_n_n.inp	Explicit dynamic continuation of sx_s_mullins.inp without importing the state or the reference configuration, C3D8RH element, cyclic uniaxial tension.
xs_s_mullins_y_y.inp	Import into Abaqus/Standard from sx_x_mullins_y_y.inp with both the state and the reference configuration imported, C3D8RH element, cyclic uniaxial tension.
xs_s_mullins_n_y.inp	Import into Abaqus/Standard from sx_x_mullins_n_y.inp with only the state imported, C3D8RH element, cyclic uniaxial tension.
xs_s_mullins_n_n.inp	Import into Abaqus/Standard from sx_x_mullins_n_n.inp without importing the state or the reference configuration, C3D8RH element, cyclic uniaxial tension.
ss_mullins_y_y.inp	Abaqus/Standard to Abaqus/Standard import from xs_s_mullins_y_y.inp with both the state and the reference configuration imported, C3D8RH element, cyclic uniaxial tension.
ss_mullins_n_y.inp	Abaqus/Standard to Abaqus/Standard import from xs_s_mullins_n_y.inp with only the state imported, C3D8RH element, cyclic uniaxial tension.

ss_mullins_n_n.inp	Abaqus/Standard to Abaqus/Standard import from xs_s_mullins_n_n.inp without importing the state or the reference configuration, C3D8RH element, cyclic uniaxial tension.
mmecdo1cut_marlow.inp	Compressible Marlow model, T2D2 element, cyclic uniaxial tension, tests temperature- and field-variable-dependent Mullins effect material properties.
mmecdo2cut_marlow.inp	Compressible Marlow model, CPS4R element, cyclic uniaxial tension, tests temperature- and field-variable-dependent Mullins effect material properties.
mmecdo3cut_marlow.inp	Compressible Marlow model, C3D8RH element, cyclic uniaxial tension.

Exponential softening

mullins_exp_ab_comp_pe.inp	Compressible Arruda-Boyce model, CPE4RH element, cyclic uniaxial tension.
mullins_exp_ogden_incomp_3d.inp	Incompressible Ogden model, C3D8RH element, cyclic uniaxial tension.
mullins_exp_user_comp_3d.inp	Compressible user-defined hyperelastic material, C3D8RH element, cyclic uniaxial tension, user subroutine *UHYPER* provided in the file `mmecdo3cut_user.f`.
mmecdo3cut_user.f	User subroutine *UHYPER* to define a compressible hyperelastic material model.
mullins_exp_vdw_comp_pe.inp	Compressible Van der Waals model, CPE4RH element, cyclic uniaxial tension.
mullins_exp_yeoh_comp_shell.inp	Compressible Yeoh model, SAX1 element, cyclic uniaxial tension.
x_mullins_exp_ab_comp_pe.inp	Explicit dynamic test with compressible Arruda-Boyce model, CPE4R element, cyclic uniaxial tension.
x_mullins_exp_ogden_comp_3d.inp	Explicit dynamic test with compressible Ogden model, C3D8R element, cyclic uniaxial tension.
x_mullins_exp_vdw_comp_pe.inp	Explicit dynamic test with compressible Van der Waals model, CPE4R element, cyclic uniaxial tension.
sx_s_mullins_exp.inp	Base problem for carrying out import from Abaqus/Standard to Abaqus/Explicit, C3D8R element, cyclic uniaxial tension.
sx_x_mullins_exp_n_y.inp	Explicit dynamic continuation of `sx_s_mullins_exp.inp` with only the state imported, C3D8R element, cyclic uniaxial tension.
xs_s_mullins_exp_n_y.inp	Abaqus/Explicit to Abaqus/Standard import from `sx_x_mullins_exp_n_y.inp` with only the state imported, C3D8R element, cyclic uniaxial tension.

Slow decay softening

mullins_slowdecay_ab_comp_pe.inp	Compressible Arruda-Boyce model, CPE4RH element, cyclic uniaxial tension.
mullins_slowdecay_ogden_incomp_3d.inp	Incompressible Ogden model, C3D8RH element, cyclic uniaxial tension.
mullins_slowdecay_user_comp_3d.inp	Compressible user-defined hyperelastic material, C3D8RH element, cyclic uniaxial tension, user subroutine *UHYPER* provided in the file `mmecdo3cut_user.f`.
mmecdo3cut_user.f	User subroutine *UHYPER* to define a compressible hyperelastic material model.
mullins_slowdecay_vdw_comp_pe.inp	Compressible Van der Waals model, CPE4RH element, cyclic uniaxial tension.
mullins_slowdecay_yeoh_comp_shell.inp	Compressible Yeoh model, SAX1 element, cyclic uniaxial tension.
x_mullins_slowdecay_ab_comp_pe.inp	Explicit dynamic test with compressible Arruda-Boyce model, CPE4R element, cyclic uniaxial tension.

x_mullins_slowdecay_ogden_comp_3d.inp	Explicit dynamic test with compressible Ogden model, C3D8R element, cyclic uniaxial tension.
x_mullins_slowdecay_vdw_comp_pe.inp	Explicit dynamic test with compressible Van der Waals model, CPE4R element, cyclic uniaxial tension.
sx_s_mullins_slowdecay.inp	Base problem for carrying out import from Abaqus/Standard to Abaqus/Explicit, C3D8R element, cyclic uniaxial tension.
sx_x_mullins_slowdecay_n_y.inp	Explicit dynamic continuation of *sx_s_mullins_slowdecay.inp* with only the state imported, C3D8R element, cyclic uniaxial tension.
xs_s_mullins_slowdecay_n_y.inp	Abaqus/Explicit to Abaqus/Standard import from *sx_x_mullins_slowdecay_n_y.inp* with only the state imported, C3D8R element, cyclic uniaxial tension.

Permanent set in elastomers

Elements tested

C3D8 C3D8H C3D8R C3D8RH CAX4R CGAX4RH CPS3 CPS4R CPS6M CPS8 S3R S4R SC8R M3D4R

Problem description

All problems in this section verify hyperelastic behavior with Mullins effect and plasticity. Comparison of finite element results can be made against the original test data (stress versus total strain) supplied with the input files. Most problems use test data as input for hyperelastic behavior and Mullins effect in a stress-free configuration. Similarly, plasticity is defined using a suitable hardening function.

The problems in this set can be broadly classified under two categories. The first category of problems consists of displacement- or load-controlled cyclic tests in modes such as uniaxial tension, biaxial tension, and simple shear with or without orientation. These problems verify simulation of permanent set with Mullins effect for various hyperelastic models.

The second category of problems is intended for testing the import capability with permanent set. Various combinations of elements and modes of deformation are verified for import from Abaqus/Standard to Abaqus/Explicit, from Abaqus/Standard to Abaqus/Standard, and from Abaqus/Explicit to Abaqus/Standard.

Material: Refer to the input files for test data and material properties used.

Loading: Both displacement- and load-controlled loading are used to verify uniaxial and biaxial tension. Only displacement-controlled loading is used to verify simple shear mode.

Results and discussion

The results of the finite element simulation can be compared with the original test data input provided in separate files, and the agreement is very good.

Input files

heplmu_matprops_calibrate.inp	Original test data that include loading and unloading data, showing permanent set in uniaxial and biaxial modes.
heplmu_matprops_bi.inp	Original test data that include loading and unloading data, showing permanent set in biaxial mode.
heplmu_matprops_uni.inp	Original test data that include loading and unloading data, showing permanent set in uniaxial mode.

heplmu_matprops.inp	Uniaxial and biaxial test data for hyperelasticity and Mullins effect in stress-free configuration, plasticity hardening data.
heplmu_matprops_bi.inp	Biaxial test data for hyperelasticity and Mullins effect in stress-free configuration, plasticity hardening data.
heplmu_matprops_uni.inp	Uniaxial test data for hyperelasticity and Mullins effect in stress-free configuration, plasticity hardening data.
heplmu_marlow_c3d8h_bi.inp	Incompressible Marlow model with biaxial test data, C3D8H element, load-controlled cyclic biaxial tension.
heplmu_marlow_c3d8rh_uniori.inp	Incompressible Marlow model with uniaxial test data, C3D8RH element with orientation, load-controlled cyclic uniaxial tension.
heplmu_ogden_biori.inp	Incompressible Ogden model with uniaxial and biaxial test data, C3D8RH element with orientation, strain-controlled cyclic biaxial tension.
heplmu_ogden_ssori.inp	Incompressible Ogden model with uniaxial and biaxial test data, C3D8H element with orientation, strain-controlled cyclic simple shear.
heplmu_ogden_uni.inp	Incompressible Ogden model with uniaxial and biaxial test data, C3D8RH element, strain-controlled cyclic uniaxial tension.
heplmu_yeoh_cgax4rh.inp	Compressible Yeoh model, CGAX4RH element, uniaxial tension followed by linear perturbation about the prestressed state.
heplmu_marlow_2d_bi.inp	Incompressible Marlow model with biaxial test data; CPS4R, CPS3, CPS6M, CPS8, S3R, S4R, SC8R, and M3D4R elements; load-controlled cyclic biaxial tension.
heplmu_marlow_2d_uniori.inp	Incompressible Marlow model with uniaxial test data; CPS3, CPS4R, CPS6M, CPS8, S3R, S4R, SC8R, and M3D4R elements with orientation; load-controlled cyclic uniaxial tension.
heplmu_ogden_2d_biori.inp	Incompressible Ogden model with uniaxial and biaxial test data; CPS3, CPS4R, CPS6M, CPS8, S3R, S4R, SC8R, and M3D4R elements with orientation; strain-controlled cyclic biaxial tension.
heplmu_ogden_2d_ssori.inp	Incompressible Ogden model with uniaxial and biaxial test data; CPS3, CPS4R, CPS6M, CPS8, S3R, and S4R elements with orientation; strain-controlled cyclic simple shear.
heplmu_ogden_2d_uni.inp	Incompressible Ogden model with uniaxial and biaxial test data; CPS3, CPS4R, CPS6M, CPS8, S3R, and S4R elements; strain-controlled cyclic uniaxial tension.
heplmu_polycomp_2d_biori.inp	Compressible polynomial model with uniaxial test data; CPS3, CPS4R, CPS6M, CPS8, S3R, S4R, SC8R, and M3D4R elements with orientation; strain-controlled cyclic biaxial tension.
x_heplmu_marlow_c3d8r_uniori.inp	Explicit dynamic test of compressible Marlow model with uniaxial test data, C3D8R element with orientation, load-controlled cyclic uniaxial tension.
x_heplmu_marlow_c3d8_bi.inp	Explicit dynamic test of compressible Marlow model with biaxial test data, C3D8 element with orientation, load-controlled cyclic biaxial tension.
x_heplmu_ogden_biori.inp	Explicit dynamic test of compressible Ogden model with uniaxial and biaxial test data, C3D8R element with orientation, strain-controlled cyclic biaxial tension.
x_heplmu_ogden_ssori.inp	Explicit dynamic test of compressible Ogden model with uniaxial and biaxial test data, C3D8 element with orientation, simple shear.
x_heplmu_ogden_uni.inp	Explicit dynamic test of compressible Ogden model with uniaxial and biaxial test data, C3D8 element, strain-controlled cyclic uniaxial tension.
x_heplmu_yeoh_cax4r.inp	Explicit dynamic test of compressible Yeoh model, CAX4R element, load-controlled cyclic uniaxial tension.

ss_s1_heplmu_uniori.inp	Base problem for Abaqus/Standard to Abaqus/Standard import, C3D8R element with compressible Marlow model in cyclic uniaxial tension.
ss_s2_heplmu_uniori.inp	Imported from ss_s1_heplmu_uniori.inp(Abaqus/Standard) with state and original reference configuration, C3D8R element with compressible Marlow model in cyclic uniaxial tension.
ss_s1_fefp2d_biori.inp	Base problem for Abaqus/Standard to Abaqus/Standard import, S4R and CPS4R elements with orientation, compressible Marlow model in cyclic biaxial tension.
ss_s2_fefp2d_biori.inp	Imported from ss_s1_fefp2d_biori.inp (Abaqus/Standard) with state and original reference configuration, S4R and CPS4R elements with orientation, compressible Marlow model in cyclic biaxial tension.
ss_s1_fefp2d_ssori.inp	Base problem for Abaqus/Standard to Abaqus/Standard import, S4R and CPS4R elements with orientation, compressible Marlow model in simple shear mode.
ss_s2_fefp2d_ssori.inp	Imported from ss_s1_fefp2d_ssori.inp (Abaqus/Standard) with state and original reference configuration, S4R and CPS4R elements with orientation, compressible Marlow model in simple shear mode.
sx_s_c3d8r_ssori.inp	Base problem for Abaqus/Standard to Abaqus/Explicit import, with subsequent import to both Abaqus/Standard and Abaqus/Explicit, C3D8R element with orientation, compressible Marlow model in simple shear mode.
sx_x_c3d8r_ssori.inp	Imported from sx_s_c3d8r_ssori.inp (Abaqus/Standard) with state and original reference configuration, base problem for Abaqus/Explicit to Abaqus/Standard import, C3D8R element with orientation, compressible Marlow model in simple shear mode.
xs_s_c3d8r_ssori.inp	Imported from sx_x_c3d8r_ssori.inp (Abaqus/Explicit) with state and original reference configuration, C3D8R element with orientation, compressible Marlow model in simple shear mode.
xx_x2_c3d8r_ssori.inp	Imported from sx_x_c3d8r_ssori.inp (Abaqus/Explicit) with state and original reference configuration, C3D8R element with orientation, compressible Marlow model in simple shear mode.
sx_s_fefp2d_biori.inp	Base problem for Abaqus/Standard to Abaqus/Explicit import, CPS4R element with orientation, compressible Marlow model in cyclic biaxial tension.
sx_x_fefp2d_biori.inp	Imported from sx_s_fefp2d_biori.inp (Abaqus/Standard) with state and original reference configuration, CPS4R element with orientation, compressible Marlow model in cyclic biaxial tension.
xs_x_heplmu_uni.inp	Base explicit dynamic problem for Abaqus/Standard to Abaqus/Explicit import, C3D8R element with compressible Marlow model and cyclic uniaxial tension.
xs_s_heplmu_uni.inp	Imported from xs_x_heplmu_uni.inp (Abaqus/Explicit) with state and original reference configuration, C3D8R element with compressible Marlow model and cyclic uniaxial tension.
xs_x_fefp2d_biori.inp	Base explicit dynamic problem for Abaqus/Explicit to Abaqus/Standard import, CPS4R element with orientation, compressible Marlow model in cyclic biaxial tension.
xs_s_fefp2d_biori.inp	Imported from xs_x_fefp2d_biori.inp (Abaqus/Explicit) with state and original reference configuration, CPS4R element with orientation, compressible Marlow model in cyclic biaxial tension.

Energy dissipation in elastomeric foams

Elements tested

CPS4R C3D8R T3D2

Problem description

The problems in this set can be broadly classified under three categories. The first category of problems consists of simple displacement- or load-controlled cyclic tests to verify the effect of energy dissipation in elastomeric foams. The tests consist of a single element that is cyclically loaded to a maximum strain (stress) level, then unloaded to zero strain (stress). This is followed by further reloading to levels of strain (stress) that are higher than those reached during the loading segment of the first cycle, followed again by unloading to zero strain (stress). The tests in this section use parts and assemblies.

The second category of problems is intended for testing the calibration capabilities for determining the Mullins effect coefficients. The problems use unloading test data that were generated by running a model with specified values of the Mullins effect coefficients. The calibration capability is meant to recover the specified values of the Mullins effect coefficients. These tests use different loading states, such as uniaxial tension, biaxial tension, and planar tension.

The third category of problems tests the import capability. All tests in this section are set up with a uniaxial stress state. The tests consist of first loading a single element in Abaqus/Standard. The results are then imported to Abaqus/Explicit, where the element is unloaded. These results are again imported back into Abaqus/Standard, where the element is loaded to deformation levels higher than the prior loading. Finally, the last set of results are imported from Abaqus/Standard to Abaqus/Standard, and then the element is unloaded. The above series of tests includes problems that import both the state and the reference configuration, problems that import only the state, and problems that import neither the state nor the reference configuration.

Material: The following material data are used for the first category of tests:

Coefficients for primary elastomeric foam behavior	Mullins effect parameters
$\mu_1 = -1048.43$, $\alpha_1 = 0.3025$, $\mu_2 = 532.20$, $\alpha_2 = 0.3958$, $\mu_3 = 517.027$, $\alpha_3 = 0.2135$, $\beta_1 = 0.2$, $\beta_2 = 0.2$, $\beta_3 = 0.2$	$r = 1.75$, $m = 0.3$, $\beta = 0.6$

Loading: The first category of problems includes both displacement- and force-controlled loading. The second and third categories of problems are carried out under only displacement-controlled loading.

Results and discussion

For the first category of problems the results of the Abaqus/Standard and Abaqus/Explicit numerical simulations are in good agreement with the analytical results.

For the second category of problems, which tests the calibration of the Mullins effect parameters, it is observed that the parameters r and β are always captured accurately. A good fit for m is obtained in situations where the deformation level leads to a relatively large value of maximum deviatoric strain energy density, U_{dev}^{m}, such that the value of βU_{dev}^{m} dominates over the value of m.

For the final category of problems, which tests the import capability, the response after each import of results is as expected. When the state is imported, further deformation on import shows the appropriate level of stress softening. On the other hand, when the state is not imported, no stress softening is observed.

Input files

mmecdo1cut_hfoam.inp	T3D2 element, cyclic uniaxial tension.
mmecdo2cut_hfoam.inp	CPS4R element, cyclic uniaxial tension.
mmecdo3cut_hfoam.inp	C3D8R element, cyclic uniaxial tension.
mmecdo3cbt_hfoam.inp	C3D8R element, cyclic biaxial tension.

mmecdo3cpt_hfoam.inp	C3D8R element, cyclic planar loading.
mmetdo3cut_hfoam.inp	Calibration test with uniaxial unloading test data, C3D8R element, cyclic uniaxial tension.
mmetdo3cbt_hfoam.inp	Calibration test with biaxial unloading test data, C3D8R element, cyclic biaxial tension.
mmetdo3cpt_hfoam.inp	Calibration test with planar unloading test data, C3D8R element, cyclic planar tension.
mmetdo3cpt_m_hfoam.inp	Calibration test with unloading test data from uniaxial, biaxial, and planar tests and with the value of the parameter *m* fixed; C3D8R element; cyclic planar tension.
x_mmecdo1cut_hfoam.inp	Explicit dynamic test, T3D2 element, cyclic uniaxial tension.
x_mmecdo2cut_hfoam.inp	Explicit dynamic test, CPS4R element, cyclic uniaxial tension.
x_mmecdo3cut_hfoam.inp	Explicit dynamic test, C3D8R element, cyclic uniaxial tension.
x_mmecdo3cbt_hfoam.inp	Explicit dynamic test, C3D8R element, cyclic biaxial tension.
x_mmecdo3cpt_hfoam.inp	Explicit dynamic test, C3D8R element, cyclic planar loading.
x_mmetdo3cut_hfoam.inp	Calibration test with uniaxial unloading test data, explicit dynamic, C3D8R element, cyclic uniaxial tension.
x_mmetdo3cbt_hfoam.inp	Calibration test with biaxial unloading test data, explicit dynamic, C3D8R element, cyclic biaxial tension.
x_mmetdo3cpt_hfoam.inp	Calibration test with planar unloading test data, explicit dynamic, C3D8R element, cyclic planar tension.
x_mmetdo3cpt_m_hfoam.inp	Calibration test with unloading test data from uniaxial, biaxial, and planar tests and with the value of the parameter *m* fixed; explicit dynamic; C3D8R element; cyclic planar tension.
x_mmecdo1cut_vishfoam.inp	Explicit dynamic test with viscoelasticity, T3D2 element, cyclic uniaxial tension.
x_mmecdo2cut_vishfoam.inp	Explicit dynamic test with viscoelasticity, CPS4R element, cyclic uniaxial tension.
x_mmecdo3cut_vishfoam.inp	Explicit dynamic test with viscoelasticity, C3D8R element, cyclic uniaxial tension.
x_mmecdo3cbt_vishfoam.inp	Explicit dynamic test with viscoelasticity, C3D8R element, cyclic biaxial tension.
x_mmecdo3cpt_vishfoam.inp	Explicit dynamic test with viscoelasticity, C3D8R element, cyclic planar loading.
x_mmetdo3cut_vishfoam.inp	Calibration test with uniaxial unloading test data, explicit dynamic with viscoelasticity, C3D8R element, cyclic uniaxial tension.
x_mmetdo3cbt_vishfoam.inp	Calibration test with biaxial unloading test data, explicit dynamic with viscoelasticity, C3D8R element, cyclic biaxial tension.
x_mmetdo3cpt_vishfoam.inp	Calibration test with planar unloading test data, explicit dynamic with viscoelasticity, C3D8R element, cyclic planar tension.
x_mmetdo3cpt_m_vishfoam.inp	Calibration test with unloading test data from uniaxial, biaxial, and planar tests and with the value of the parameter *m* fixed; explicit dynamic with viscoelasticity; C3D8R element; cyclic planar tension.
sx_s_mullins_hfoam.inp	Base problem for carrying out import from Abaqus/Standard to Abaqus/Explicit, C3D8R element, cyclic uniaxial tension.
sx_x_mullins_hfoam_n_y.inp	Explicit dynamic continuation of sx_s_mullins_hfoam.inp with only the state imported, C3D8R element, cyclic uniaxial tension.

xs_s_mullins_hfoam_n_y.inp	Import into Abaqus/Standard from sx_x_mullins_hfoam_n_y.inp with only the state imported, C3D8R element, cyclic uniaxial tension.
ss_mullins_hfoam_n_y.inp	Abaqus/Standard to Abaqus/Standard import from xs_s_mullins_hfoam_n_y.inp with only the state imported, C3D8R element, cyclic uniaxial tension.

Initial stress for materials with Mullins effect

Elements tested

C3D8R S4R

Problem description

These tests verify if the initial stresses are applied correctly for materials with a Mullins effect model. The input files consist of base models that are run first, and the models with initial conditions use the results from the base model to prescribe initial stresses and initial maximum energy densities. In the second input files the initial conditions are specified using result values from a point on the loading path in the first analysis or on the unloading path in the first analysis. Prescribing the initial maximum energy density in addition to initial stresses allows Abaqus to compute the correct state accounting for Mullins effect. In the second analysis the same loads are applied as in the base analysis. Therefore, if the initial state in the second analysis is set correctly, both analyses produce the same results.

Material 1:

Neo Hookean coefficients: C_{10}=80.0, D_1= 2.013423× 10^{-4}
Ogden-Roxburgh coefficients: r=1.5, m=10.0, β=1.0

Material 2:

Ogden coefficients: μ_1=160.0, α_1=2.0, D_1= 2.013423× 10^{-4}
Exponential model coefficients: η_F=0.4, U_h=10.0, α=1.0
Prony series coefficients (N=2): $\overline{g}_1^P$ = 0.31, $\overline{k}_1^P$ = 0.2, τ_1=2.0× 10^6, $\overline{g}_2^P$ = 0.19, $\overline{k}_2^P$ = 0.1, τ_2=1.0× 10^5

Results and discussion

The results from the base analyses agree well with the results from the second analyses with initial conditions.

Input files

neoh_ogden_roxburgh_cycle_c3d8r.inp	Standard analysis, compressible Neo Hookean model, Ogden-Roxburgh model, C3D8R element.
neoh_ogden_roxburgh_cycle_c3d8r_load_ic.inp	Standard analysis, compressible Neo Hookean model, Ogden-Roxburgh model, C3D8R element, initial conditions specified using result values from a point on the loading path in the first analysis.
neoh_ogden_roxburgh_cycle_c3d8r_unload_ic.inp	Standard analysis, compressible Neo Hookean model, Ogden-Roxburgh model, C3D8R element, initial conditions specified using result values from a point on the unloading path in the first analysis.

ogden_ve_exp_cycle_c3d8r.inp	Standard analysis, compressible Ogden model, exponential model, viscoelasticity, C3D8R element.
ogden_ve_exp_cycle_c3d8r_load_ic.inp	Standard analysis, compressible Ogden model, exponential model, viscoelasticity, C3D8R element, initial conditions specified using result values from a point on the loading path in the first analysis.
ogden_ve_exp_cycle_c3d8r_unload_ic.inp	Standard analysis, compressible Ogden model, exponential model, viscoelasticity, C3D8R element, initial conditions specified using result values from a point on the unloading path in the first analysis.
ogden_ve_exp_cycle_s4r.inp	Standard analysis, incompressible Ogden model, exponential model, viscoelasticity, S4R element.
ogden_ve_exp_cycle_s4r_load_ic.inp	Standard analysis, incompressible Ogden model, exponential model, viscoelasticity, S4R element, initial conditions specified using result values from a point on the loading path in the first analysis.
ogden_ve_exp_cycle_s4r_unload_ic.inp	Standard analysis, incompressible Ogden model, exponential model, viscoelasticity, S4R element, initial conditions specified using result values from a point on the unloading path in the first analysis.
x_neoh_ogden_roxburgh_cycle_c3d8r_unload_ic.inp	Explicit dynamic analysis, compressible Neo Hookean model, Ogden-Roxburgh model, C3D8R element, initial conditions specified using result values from a point on the unloading path in the first analysis.
x_ogden_ve_exp_cycle_c3d8r_load_ic.inp	Explicit dynamic analysis, compressible Ogden model, exponential model, viscoelasticity, C3D8R element, initial conditions specified using result values from a point on the loading path in the first analysis.
x_ogden_ve_exp_cycle_s4r.inp	Explicit dynamic analysis, Ogden model, exponential model, viscoelasticity, S4R element.
x_ogden_ve_exp_cycle_s4r_load_ic.inp	Explicit dynamic analysis, Ogden model, exponential model, viscoelasticity, S4R element, initial conditions specified using result values from a point on the loading path in the first analysis.
x_ogden_ve_exp_cycle_s4r_unload_ic.inp	Explicit dynamic analysis, Ogden model, exponential model, viscoelasticity, S4R element, initial conditions specified using result values from a point on the unloading path in the first analysis.

Figures

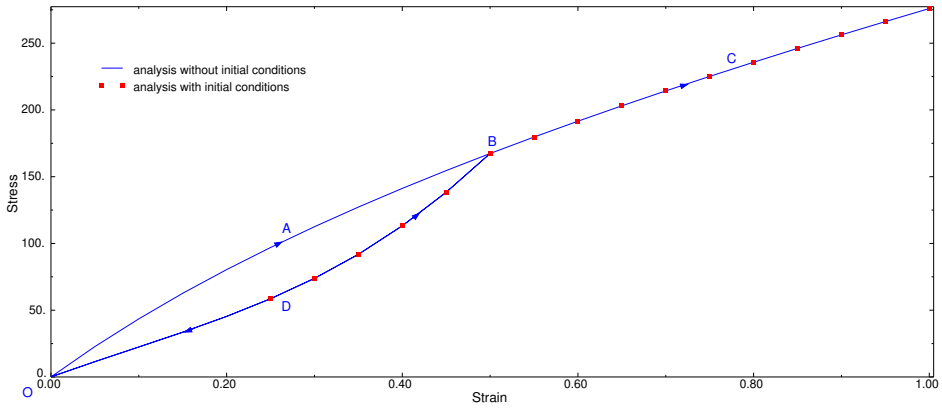


Figure 1: Comparison of the stress-strain responses of the base analysis and the analysis with initial conditions for material using the Neo Hookean model with the Ogden-Roxburgh Mullins effect model.

Hysteretic materials

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Bergström-Boyce hysteresis for elastomers: test of time-dependent behavior

Elements tested

CAX4 CPE4

Problem description

The problems in this set are simulations of experiments presented in *Bergström and Boyce (1998)*. The Abaqus/Standard results are compared to the Bergström and Boyce results.

The tests consist of uniaxial compression of disk-like rubber specimens (height = 13 mm, diameter = 28 mm) and plane strain compression of rectangular specimens (height = 13 mm, cross-sectional area = 140 mm^2). The materials used in the tests are Chloroprene rubber with varying carbon black filler concentrations and unfilled Nitrile rubber. The specimens are subjected to constant strain rate, cyclic loading, and constant strain-rate load cycles interspersed with relaxation segments of varying time intervals. The strain measure used here refers to logarithmic strain.

Two problems that test the creep strain-rate regularizing parameter, E, have also been included.

Material:

Arruda-Boyce hyperelasticity

Chloroprene rubber (15 pph. carbon black)

μ = 0.6 MPa, $\lambda_m = \sqrt{8}$ = 2.8284, D = 0.01

Chloroprene rubber (40 pph. carbon black)

μ = 1.08 MPa, $\lambda_m = \sqrt{8}$ = 2.8284, D = 0.01

Chloroprene rubber (65 pph. carbon black)

μ = 1.71 MPa, $\lambda_m = \sqrt{8}$ = 2.8284, D = 0.01

Nitrile rubber (unfilled)

μ = 0.87 MPa, $\lambda_m = \sqrt{6}$ = 2.4495, D = 0.01

Hysteresis

Chloroprene rubber (15 pph. carbon black)

S = 1.6, A = 0.9526 $(MPa)^{-4}\ s^{-1}$, m = 4.0, C = −1.0

Chloroprene rubber (40 pph. carbon black)

$S = 2.0$, $A = 0.9526$ $(\text{MPa})^{-4}\,\text{s}^{-1}$, $m = 4.0$, $C = -1.0$

Chloroprene rubber (65 pph. carbon black)

$S = 4.0$, $A = 0.03135$ $(\text{MPa})^{-5}\,\text{s}^{-1}$, $m = 5.0$, $C = -0.9$

Nitrile rubber (unfilled)

$S = 2.5$, $A = 0.5500$ $(\text{MPa})^{-5}\,\text{s}^{-1}$, $m = 5.0$, $C = -0.6$

Loading: The loading is imposed with displacement-controlled boundary conditions.

Results and discussion

The results of the Abaqus/Standard numerical simulations are in very good agreement with the results presented in *Bergström and Boyce (1998)*. The results for the problems that test the creep strain-rate regularizing parameter, E, are almost identical to the results without the use of this parameter in all regions except around zero strain, where the results with this parameter are smoother.

Input files

mbbcdo3euc_un_1e_2_cl15.inp	Uniaxial cyclic compression, linear perturbation with **LOAD CASE*, CAX4 elements; repeated cycling; strain rate = −0.01/s; Chloroprene rubber (15 pph. carbon black).
mbbcdo3mcy_ps_1e_2_cl65.inp	Plane strain cyclic compression, CPE4 elements; strain rate = −0.01/s; Chloroprene rubber (65 pph. carbon black).
mbbcdo3mcy_un_1e_2_cl15.inp	Uniaxial cyclic compression, CAX4 elements; strain rate = −0.01/s; Chloroprene rubber (15 pph. carbon black).
mbbcdo3mcy_un_1e_2_cl40.inp	Uniaxial cyclic compression, CAX4 elements; strain rate = −0.01/s; Chloroprene rubber (40 pph. carbon black).
mbbcdo3mcy_un_1e_2_cl65.inp	Uniaxial cyclic compression, CAX4 elements; strain rate = −0.01/s; Chloroprene rubber (65 pph. carbon black).
mbbcdo3mcy_un_1e_2_ni.inp	Uniaxial cyclic compression, CAX4 elements; strain rate = −0.01/s; Nitrile rubber (unfilled).
mbbcdo3mcy_un_23e_5_ni.inp	Uniaxial cyclic compression, CAX4 elements; strain rate = −0.00023/s; Nitrile rubber (unfilled).
mbbcdo3rcy_un_1e_1_cl15.inp	Uniaxial cyclic compression with 1 relaxation segment in both uploading and unloading; relaxation strain = −0.6; strain rate = −0.1/s; relaxation time = 1000s; Chloroprene rubber (15 pph. carbon black).
mbbcdo3ruc_un_1e_1_cl15.inp	Uniaxial cyclic compression with two relaxation segments in both uploading and unloading; relaxation strains = −0.26 and −0.54; strain rate = −0.1/s; relaxation time = 30s; Chloroprene rubber (15 pph. carbon black).
mbbcdo3rcy_un_2e_3_cl15.inp	Uniaxial cyclic compression with two relaxation segments in both uploading and unloading; relaxation strains = −0.3 and −0.6; strain rate = −0.002/s; relaxation time = 120s; Chloroprene rubber (15 pph. carbon black).
hysteresis_e001_uniaxial.inp	Cyclic uniaxial straining; test for creep strain-rate regularizing parameter E: CAX4 element.
hysteresis_e001_biaxial.inp	Cyclic biaxial straining; test for creep strain-rate regularizing parameter E: C3D8R element.

Bergström-Boyce hysteresis for elastomers: test of time-independent behavior

Elements tested

C3D8 C3D8H C3D8R C3D8RH CPE4

Problem description

The problems in this set test and verify the performance of the hysteresis material model in conjunction with some of the hyperelastic potentials available in Abaqus/Standard. The problems involve imposing homogeneous/inhomogeneous deformations over very short periods of time in comparison with the characteristic relaxation time of the hysteresis model. Since the stress-scaling factor is taken to be 1.0 for all the tests, the stresses of this step should be very close to twice the values obtained from running the corresponding problems without the hysteresis option but with the same hyperelastic material definition. In a second step the boundary conditions are held fixed and the stresses are allowed to relax. The stresses at the end of this step from the hysteresis calculations should be close to the values obtained in the run with the hyperelastic material.

For the test with hydrostatic compression loading, *mbbcdo3ahc.inp*, and the uniaxial loading test, *mbbtdo3hut.inp*, the stresses obtained should be twice those obtained in the corresponding problems run solely with hyperelasticity. This is a consequence of the fact that, in the test with hydrostatic compression loading, the induced stresses are purely hydrostatic; such a stress state is incapable of inducing inelastic deformation in the material model. The uniaxial loading test involves a creep constant of $A = 0.0$, which is equivalent to eliminating the creep response of the model. The factor of 2 in the stress output is a result of the choice of the stress scaling factor, $S = 1$. These two problems are run as single-step analyses.

In the problems that use reduced-integration elements, the hourglass stiffness is verified as being calculated on the basis of the instantaneous moduli.

A single problem also verifies using a hyperelastic material with an instantaneous elastic moduli definition and hysteresis effects. The elastic constants in the file *mbbcot3hut_inst.inp* are taken to be $(1 + S)$ times the constants of the corresponding problem with the default long-term moduli specification, *mbbcot3hut.inp*. (The constant corresponding to the volumetric part of the strain energy, D_i, should be divided by the same factor; in this problem it is of no consequence since the material is completely incompressible.) The results of these two problems are verified to be identical.

The problem *mbbcoo3vlp.inp* tests linear perturbation results. A purely hyperelastic response is recovered in this analysis by setting the creep scaling parameter to 0.0, which facilitates comparison with the identical problem run with only hyperelastic behavior (*mhecoo3vlp.inp*).

Results and discussion

All tests yield the expected results, as defined in the problem description.

Input files

mbbcdo3ahc.inp	Compressible, polynomial (N=1), hydrostatic compression, C3D8 elements.
mbbcdo3gsh_ogden.inp	Compressible, Ogden (N=1), nonuniform shear, C3D8R elements.
mbbcdo3gsh_redpol.inp	Compressible, reduced polynomial (N=1), nonuniform shear, C3D8 elements.
mbbcdo3gsh_vwaals.inp	Compressible, Van der Waals, nonuniform shear, C3D8 elements.
mbbcdo3gsh_yeoh.inp	Compressible, Yeoh, nonuniform shear, C3D8 elements.
mbbcdo3ibt.inp	Compressible, Arruda-Boyce, biaxial tension, linear perturbation with **LOAD CASE*, CPE4 elements.
mbbcoo3hut.inp	Incompressible, polynomial (N=1), uniaxial tension, C3D8H elements.

mbbtdo3hut.inp	Compressible, polynomial (N=1), test data, uniaxial tension, C3D8 elements.
mbbcot3hut.inp	Incompressible, temperature-dependent elasticity, polynomial (N=1), uniaxial tension, C3D8RH elements.
mbbcot3hut_inst.inp	Incompressible, temperature-dependent instantaneous elasticity, polynomial (N=1), uniaxial tension, C3D8RH elements.
mbbcoo3vlp.inp	Incompressible, uniaxial tension, static linear perturbation steps, C3D8RH elements.

References

- Bergström, J. S., and M. C. Boyce, "Constitutive Modeling of the Large Strain Time-Dependent Behavior of Elastomers," *Journal of the Mechanics and Physics of Solids*, vol. 46, pp. 931–954, 1998.

Temperature-dependent elastic materials

Products: Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

T2D2 T3D2 B21 B31 PIPE21 PIPE31 SAX1 S4R S4RS S4RSW C3D8R C3D10M CPE4R CPE6M CPS4R CPS6M CAX4R CAX6M M3D4R

Features tested

Temperature-dependent material properties with predefined temperature fields are tested for the following elastic material models: isotropic elasticity, orthotropic elasticity, anisotropic elasticity, and lamina.

Problem description

This verification test consists of a set of single element models that include combinations of all the available element types with all the available material models. All the elements are loaded with a tensile load defined by specifying the vertical velocity at the top nodes of each element with the bottom nodes fixed. The velocity is ramped from zero to 0.2. The temperature at all nodes increases from an initial value of 0° to a final value of 100°. The material properties are defined as a linear function of temperature, as shown in *Table 1*. The density for all the materials is 7850. For every material model, only those element types available for the model are used. The undeformed meshes are shown in *Figure 1*.

Results and discussion

Figure 2 shows the plot of vertical stress versus vertical strain for the isotropic elasticity model. The plots of vertical stress versus vertical strain for orthotropic elasticity (ENGINEERING CONSTANTS), orthotropic elasticity (ORTHOTROPIC), anisotropic elasticity, and lamina are shown in *Figure 3*, *Figure 4*, *Figure 5*, and *Figure 6*, respectively. The vertical stress and vertical strain are σ_{11} and ϵ_{11} for the truss, beam, and axisymmetric shell elements and σ_{22} and ϵ_{22} for the remaining elements. The results from pipe elements are consistent with the beams.

Input files

temp_elastic.inp	Input data used in this analysis.
temp_elastic_ef1.inp	External file referenced in this analysis.
temp_elastic_simpson.inp	Explicit dynamics analysis using Simpson integration through the shell thickness.
temp_elastic_restart.inp	Restart data that completes 25 milliseconds of the response.

Tables

Table 1: Material properties.

Material	Properties	T=0	T=100
Isotropic elasticity	E	193.1×10^9	97.0×10^9
	ν	0.0	0.0
Orthotropic elasticity	E_1	2.0×10^{11}	1.0×10^{11}
(ENGINEERING CONSTANTS)	E_2	1.0×10^{11}	5.0×10^{10}
	E_3	1.0×10^{11}	5.0×10^{10}
	ν_{12}	0.0	0.0
	ν_{13}	0.0	0.0
	ν_{23}	0.0	0.0
	G_{12}	7.69×10^{10}	6.69×10^{10}
	G_{13}	7.69×10^{10}	6.69×10^{10}
	G_{23}	9.0×10^9	8.0×10^9
Orthotropic elasticity	D_{1111}	2.24×10^{11}	1.00×10^{11}
(ORTHOTROPIC)	D_{1122}	4.79×10^5	4.59×10^5
	D_{2222}	1.23×10^{11}	0.5×10^{11}
	D_{1133}	4.21×10^5	4.00×10^5
	D_{2233}	4.74×10^5	4.00×10^5
	D_{3333}	1.21×10^{11}	0.5×10^{11}
	D_{1212}	7.69×10^{10}	7.00×10^{10}
	D_{1313}	7.69×10^{10}	7.00×10^{10}
	D_{2323}	9.00×10^9	8.00×10^9
Lamina	E_1	2.0×10^{11}	1.0×10^{11}
	E_2	1.5×10^{11}	0.7×10^{11}
	ν_{12}	0.0	0.0
	G_{12}	2.00×10^{10}	1.80×10^{10}
	G_{13}	9.00×10^9	8.00×10^9
	G_{23}	8.50×10^9	7.50×10^9

Material	Properties	T=0	T=100
Anisotropic elasticity	D_{1111}	2.24×10^{11}	1.00×10^{11}
	D_{1122}	4.79×10^{5}	4.00×10^{5}
	D_{2222}	1.23×10^{11}	0.5×10^{11}
	D_{1133}	4.21×10^{5}	4.00×10^{5}
	D_{2233}	4.74×10^{5}	4.00×10^{5}
	D_{3333}	1.21×10^{11}	0.5×10^{11}
	D_{1112}	1.00×10^{6}	9.00×10^{5}
	D_{2212}	2.00×10^{6}	1.80×10^{6}
	D_{3312}	3.00×10^{6}	2.60×10^{6}
	D_{1212}	7.69×10^{10}	7.00×10^{10}
	D_{1113}	4.00×10^{6}	3.60×10^{6}
	D_{2213}	5.00×10^{6}	4.60×10^{6}
	D_{3313}	6.00×10^{6}	5.60×10^{6}
	D_{1213}	7.00×10^{6}	6.60×10^{6}
	D_{1313}	7.69×10^{10}	7.00×10^{10}
	D_{1123}	8.00×10^{6}	7.60×10^{6}
	D_{2223}	9.00×10^{6}	8.00×10^{6}
	D_{3323}	1.00×10^{7}	9.00×10^{6}
	D_{1223}	1.10×10^{7}	1.00×10^{7}
	D_{1323}	1.20×10^{7}	1.10×10^{7}
	D_{2323}	9.00×10^{9}	8.00×10^{9}

Figures

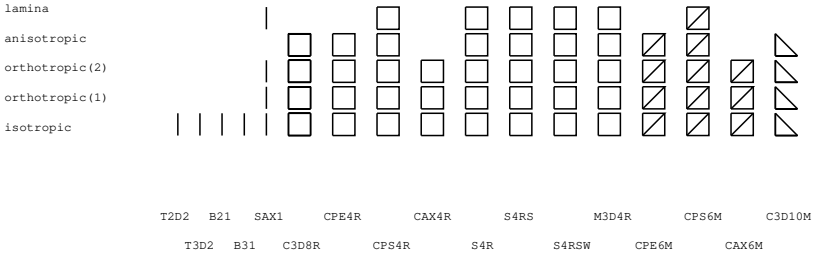


Figure 1: Temperature-dependent material property test for elastic materials.

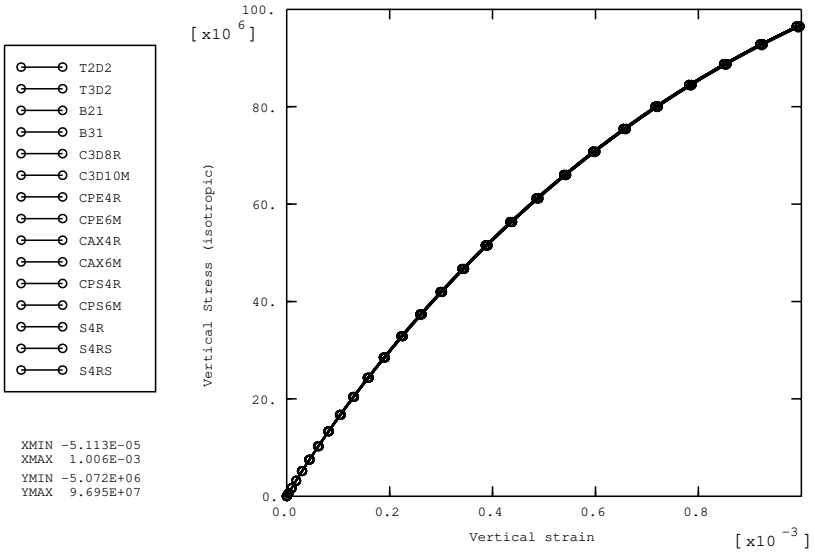


Figure 2: Vertical stress versus vertical strain for isotropic elasticity.

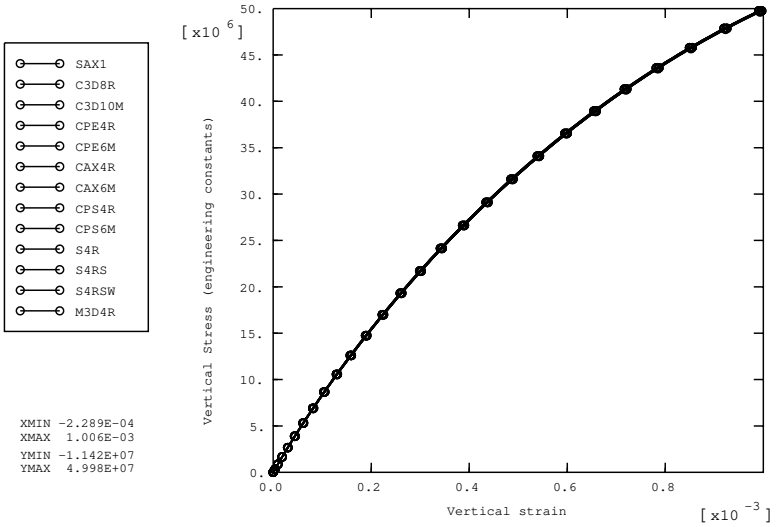


Figure 3: Vertical stress versus vertical strain for orthotropic elasticity (ENGINEERING CONSTANTS).

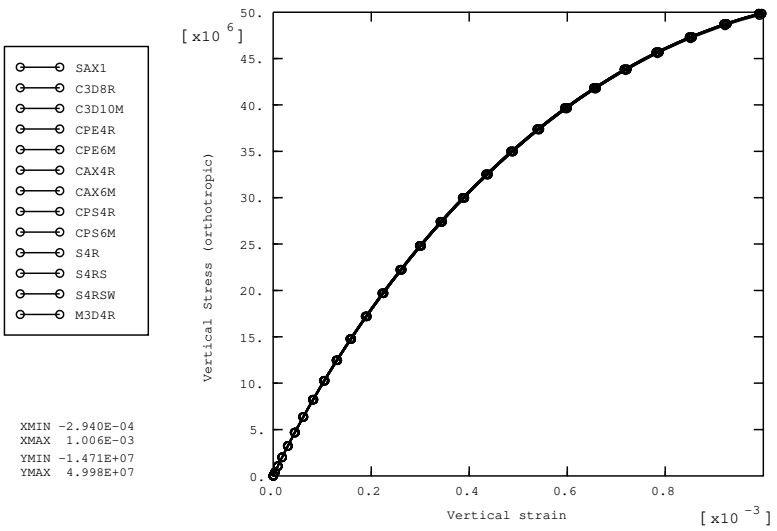


Figure 4: Vertical stress versus vertical strain for orthotropic elasticity (ORTHOTROPIC).

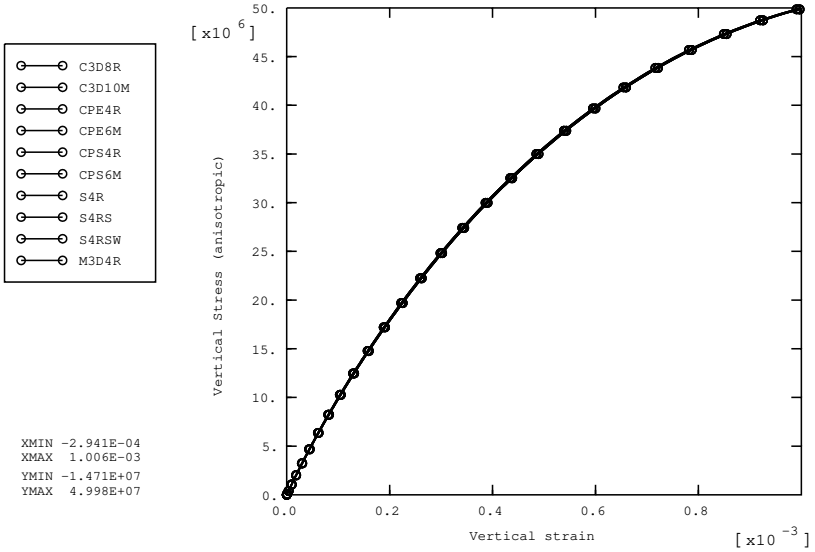


Figure 5: Vertical stress versus vertical strain for anisotropic elasticity.

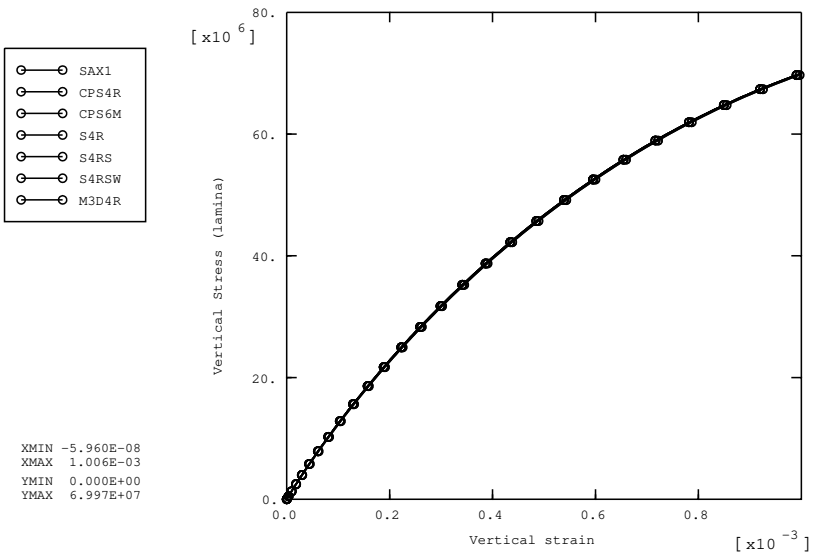


Figure 6: Vertical stress versus vertical strain for lamina.

Field-variable-dependent elastic materials

Products: Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

T2D2 T3D2 B21 B31 PIPE21 PIPE31 SAX1 S4R S4RS S4RSW C3D8R C3D10M CPE4R CPE6M CPS4R CPS6M CAX4R CAX6M M3D4R

Features tested

Field-variable-dependent material properties with predefined field variables are tested for the following elastic material models: isotropic elasticity, orthotropic elasticity, anisotropic elasticity, and lamina.

Problem description

This verification test consists of a set of single element models that include combinations of all the available element types with all the available material models. All elements are loaded with a tensile load defined by specifying the vertical velocity at the top nodes of each element with the bottom nodes fixed. The velocity is ramped from zero to 0.2. One field variable, which increases from an initial value of 0 to a final value of 100, is defined at all the nodes. Material properties are defined as a linear function of the field variable, shown in *Table 1*. The density for all the materials is 7850. For every material model only those element types available for the model are used. The undeformed meshes are shown in *Figure 1*.

Results and discussion

Figure 2 shows the plot of vertical stress versus vertical strain for the isotropic elasticity model. The plots of vertical stress versus vertical strain for orthotropic elasticity (ENGINEERING CONSTANTS), orthotropic elasticity (ORTHOTROPIC), anisotropic elasticity, and lamina are shown in *Figure 3*, *Figure 4*, *Figure 5*, and *Figure 6*, respectively. The vertical stress and vertical strain are σ_{11} and ϵ_{11} for the truss, beam, and axisymmetric shell elements and σ_{22} and ϵ_{22} for the remaining elements. The results from pipe elements are consistent with the beams.

Input files

field_elastic.inp	Input data used in this analysis.
field_elastic_ef1.inp	External file referenced in this input.

Tables

Table 1: Material properties.

Material	Properties	fv=0	fv=100
Isotropic elasticity	E	193.1×10^9	97×10^9

Material	Properties	fv=0	fv=100
	ν	0.0	0.0
Orthotropic elasticity	E_1	2.0×10^{11}	1.0×10^{11}
(ENGINEERING CONSTANTS)	E_2	1.0×10^{11}	5.0×10^{10}
	E_3	1.0×10^{11}	5.0×10^{10}
	ν_{12}	0.0	0.0
	ν_{13}	0.0	0.0
	ν_{23}	0.0	0.0
	G_{12}	7.69×10^{10}	6.69×10^{10}
	G_{13}	7.69×10^{10}	6.69×10^{10}
	G_{23}	9.0×10^{9}	8.0×10^{9}
Orthotropic elasticity	D_{1111}	2.24×10^{11}	1.00×10^{11}
(ORTHOTROPIC)	D_{1122}	4.79×10^{5}	4.59×10^{5}
	D_{2222}	1.23×10^{11}	0.5×10^{11}
	D_{1133}	4.21×10^{5}	4.00×10^{5}
	D_{2233}	4.74×10^{5}	4.00×10^{5}
	D_{3333}	1.21×10^{11}	0.5×10^{11}
	D_{1212}	7.69×10^{10}	7.00×10^{10}
	D_{1313}	7.69×10^{10}	7.00×10^{10}
	D_{2323}	9.00×10^{9}	8.00×10^{9}
Lamina	E_1	2.0×10^{11}	1.0×10^{11}
	E_2	1.5×10^{11}	0.7×10^{11}
	ν_{12}	0.0	0.0
	G_{12}	2.00×10^{10}	1.80×10^{10}
	G_{13}	9.00×10^{9}	8.00×10^{9}
	G_{23}	8.50×10^{9}	7.50×10^{9}
Anisotropic elasticity	D_{1111}	2.24×10^{11}	1.00×10^{11}
	D_{1122}	4.79×10^{5}	4.00×10^{5}
	D_{2222}	1.23×10^{11}	0.5×10^{11}
	D_{1133}	4.21×10^{5}	4.00×10^{5}

Material	Properties	fv=0	fv=100
	D_{2233}	4.74×10^5	4.00×10^5
	D_{3333}	1.21×10^{11}	0.5×10^{11}
	D_{1112}	1.00×10^6	9.00×10^5
	D_{2212}	2.00×10^6	1.80×10^6
	D_{3312}	3.00×10^6	2.60×10^6
	D_{1212}	7.69×10^{10}	7.00×10^{10}
	D_{1113}	4.00×10^6	3.60×10^6
	D_{2213}	5.00×10^6	4.60×10^6
	D_{3313}	6.00×10^6	5.60×10^6
	D_{1213}	7.00×10^6	6.60×10^6
	D_{1313}	7.69×10^{10}	7.00×10^{10}
	D_{1123}	8.00×10^6	7.60×10^6
	D_{2223}	9.00×10^6	8.00×10^6
	D_{3323}	1.00×10^7	9.00×10^6
	D_{1223}	1.10×10^7	1.00×10^7
	D_{1323}	1.20×10^7	1.10×10^7
	D_{2323}	9.00×10^9	8.00×10^9

Figures

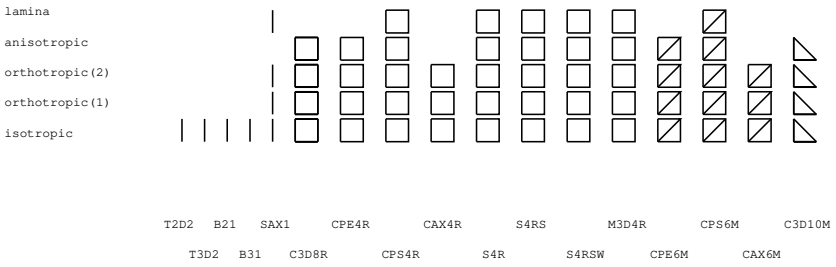


Figure 1: Field-variable-dependent material property test for elastic materials.

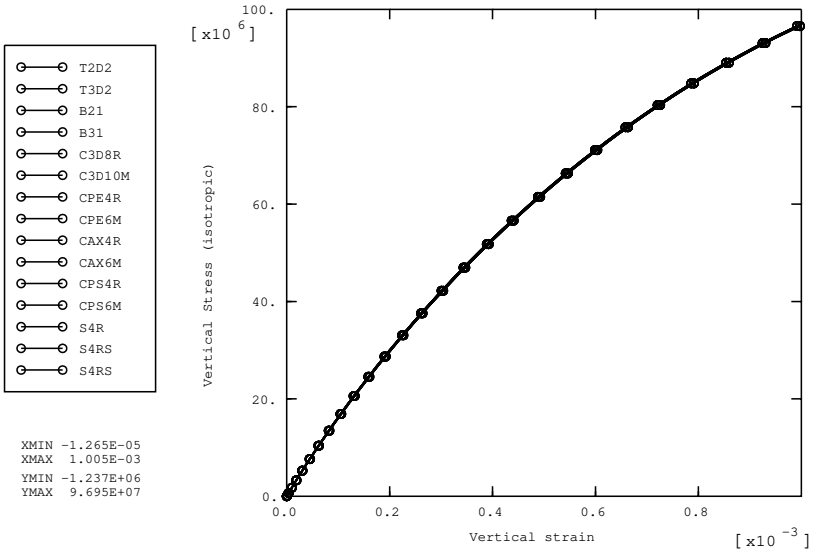


Figure 2: Vertical stress vs. vertical strain for isotropic elasticity.

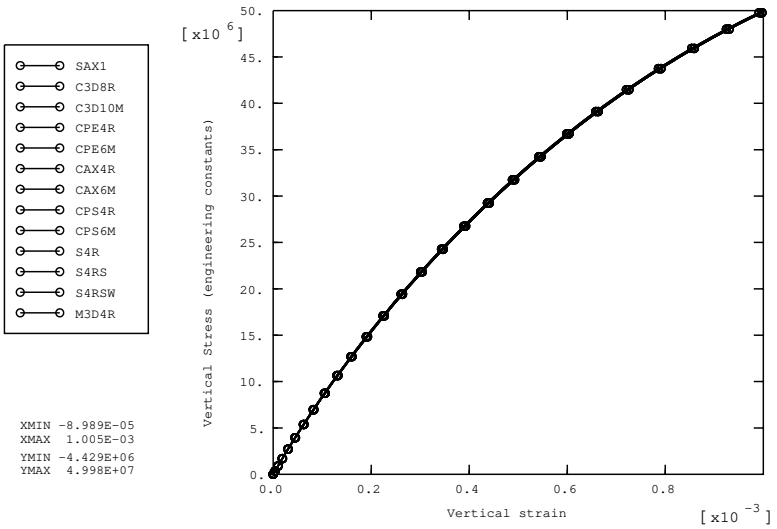


Figure 3: Vertical stress vs. vertical strain for orthotropic elasticity (ENGINEERING CONSTANTS).

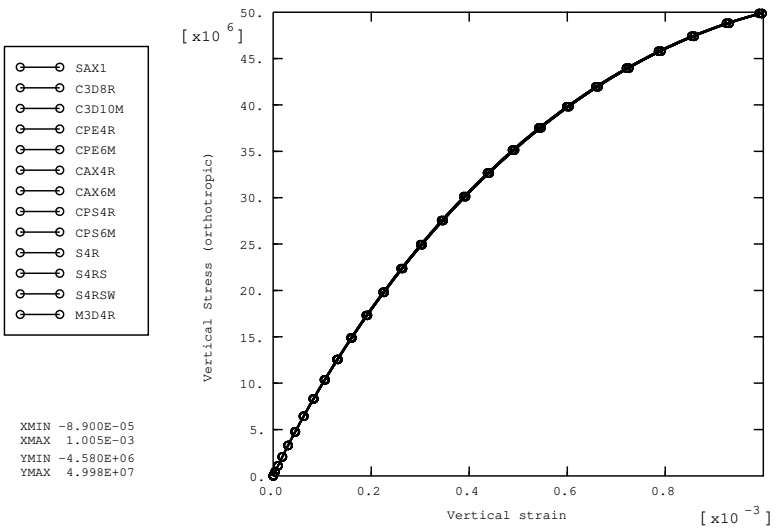


Figure 4: Vertical stress vs. vertical strain for orthotropic elasticity (ORTHOTROPIC).

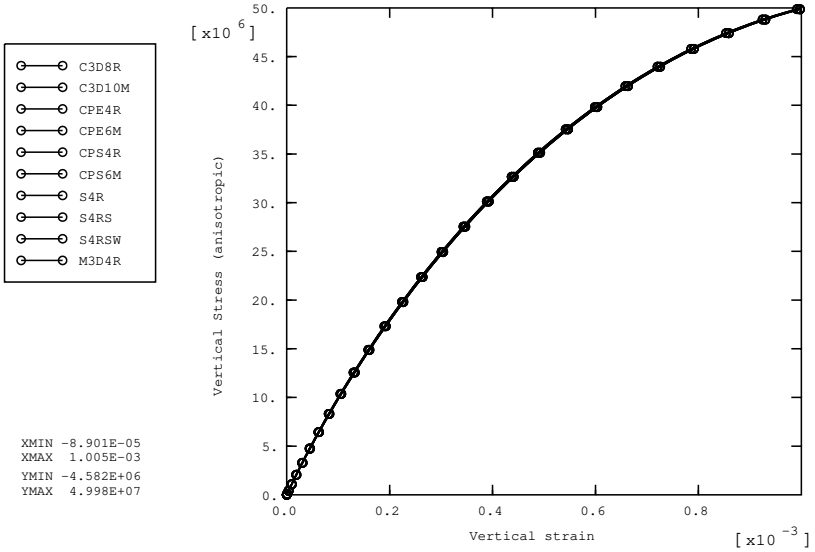


Figure 5: Vertical stress vs. vertical strain for anisotropic elasticity.

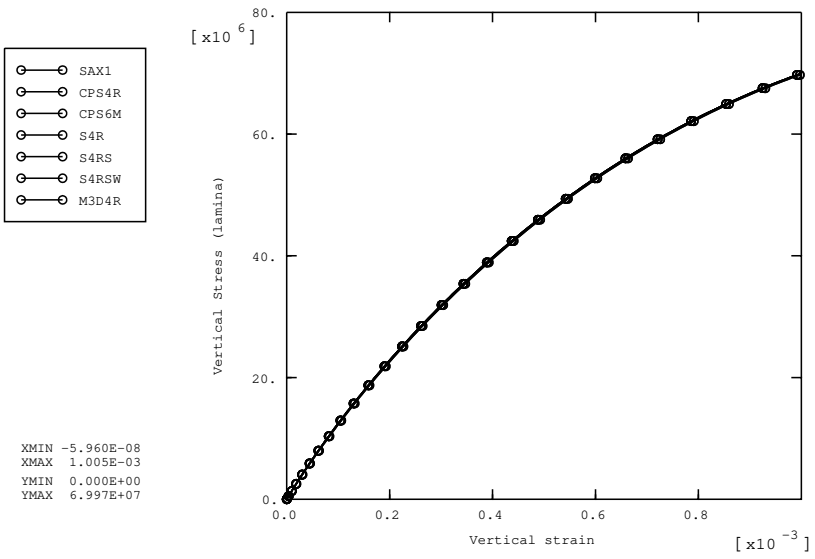


Figure 6: Vertical stress vs. vertical strain for lamina.

Large-strain viscoelasticity with hyperelasticity

Products: Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

CAX4R CPE4R CPS4R M3D4R

Features tested

Large deformation kinematics, viscoelastic-hyperelastic material model.

Problem description

This example is used to verify the viscoelastic material model in Abaqus/Explicit. In all of the problems, the material is defined by the hyperelastic polynomial formulation with $N = 1$. The viscoelastic behavior is given by Prony series parameters or by using the test data input option.

The first test is volumetric relaxation. A single element is compressed at a uniform rate over a period of time during which the stresses are allowed to relax. This problem tests that the volumetric relaxation behavior is captured correctly. Plane stress and plane strain elements are used in this test.

The second test is uniaxial relaxation. A single element is stretched at a uniform rate over a period of time during which the stresses are allowed to relax. This problem verifies that the shear relaxation behavior is captured correctly. Plane strain and membrane elements are used in this test.

The third test is circumferential relaxation. All nodes of a single axisymmetric element are moved radially outward at a uniform rate over a period of time during which the stresses are allowed to relax. All the nodes are fixed in the axial direction. This problem verifies that the shear and volumetric relaxation behavior is correct in the circumferential direction.

Results and discussion

The time histories of the stresses are shown in *Figure 1* through *Figure 3*. *Figure 1* shows the volumetric response of the material with viscoelastic properties compared to the response without viscoelastic properties. *Figure 2* shows the tensile response of the material with viscoelastic properties compared to the response without viscoelastic properties. *Figure 3* shows the circumferential response of the material with viscoelastic properties compared to the response without viscoelastic properties.

This problem tests the features listed but does not provide independent verification of them.

Input files

visco_vol_pe.inp	Volumetric relaxation test using plane strain elements.
visco_uni_mem.inp	Uniaxial relaxation test using membrane elements.
visco_circum_axi.inp	Circumferential creep test using axisymmetric elements.
visco_vol_ps.inp	Volumetric relaxation test using plane stress elements.

visco_uni_pe.inp	Uniaxial relaxation test using plane strain elements.
visco_uni_pe_frq.inp	Uniaxial relaxation test using plane strain elements and using Prony parameters calibrated from frequency-dependent moduli data.

Figures

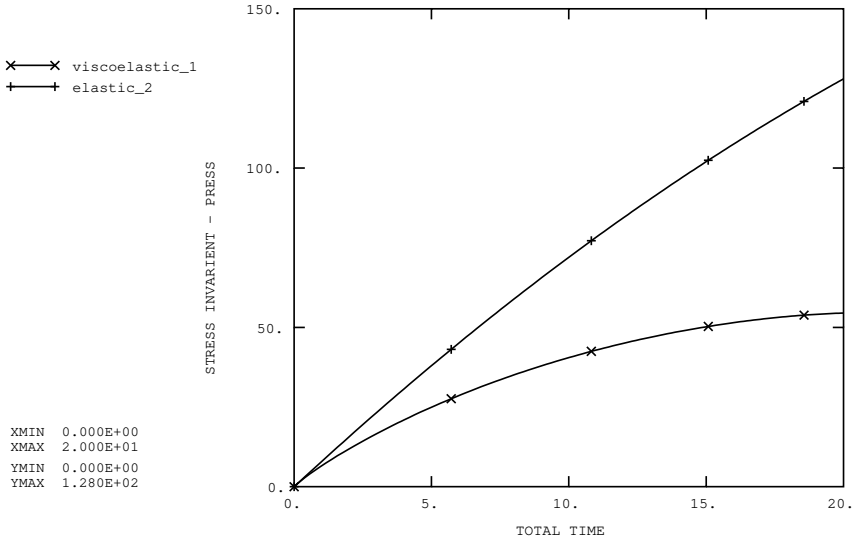


Figure 1: Pressure stress versus time for volumetric compression (CPE4R).

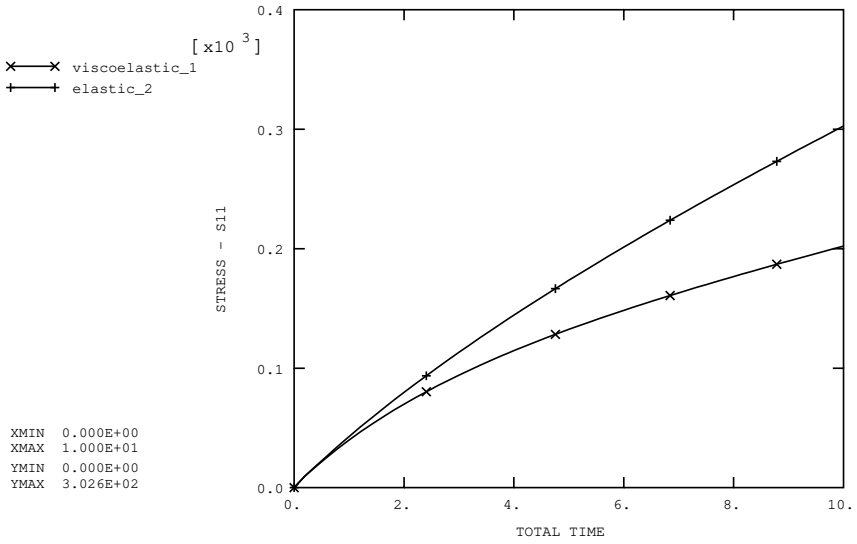


Figure 2: Tensile stress versus time for uniaxial tension (M3D4R).

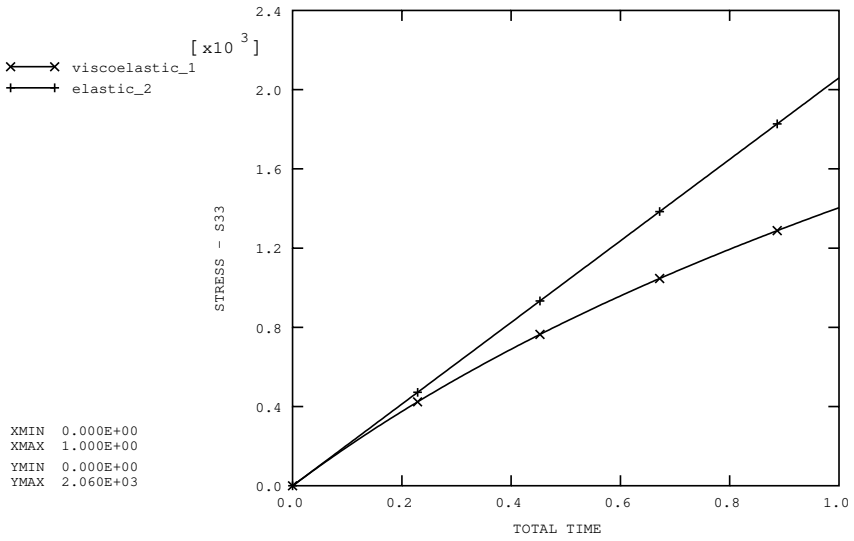


Figure 3: Circumferential stress versus time for the circumferential expansion (CAX4R).

Nonlinear large-strain viscoelasticity with hyperelasticity

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D8 C3D8H C3D8R C3D8RH
CAX4 CAX4H CAX4R CAX4RH
CPE4 CPE4H CPE4R CPE4RH

Features tested

Nonlinear viscoelastic material model.

Problem description

In each input file a single element is subjected to a uniaxial load. The input files consist of two steps. In the first step the load is gradually increased until it reaches the prescribed value. In the second step the value of the load is kept constant for a specified time. These tests verify the correctness of the creep behavior with different element types.

Material: The nonlinear viscoelastic material consists of three or four networks, including a purely elastic network. The strain-hardening power law, the hyperbolic-sine, the Bergstrom-Boyce, the Darveaux, and the modified Darveaux creep models are used to model the viscoelastic behavior. Creep model coefficients are shown in *Table 1*, *Table 2*, *Table 3*, *Table 4*, and *Table 5*. Several different hyperelastic materials are tested, as described in *Table 6*.

Viscous softening is also tested, both in exponential form with the coefficients shown in *Table 7* and in an equivalent tabular form. An equivalent implementation using user subroutines *UHYPER* and *VUHYPER* is provided for comparison and validation purposes.

Table 1: Strain-hardening power law creep model coefficients.

$A=10^{-5}$
$n=1.0$
$m=-0.5$

Table 2: Hyperbolic-sine creep model coefficients.

$A=2.5\times10^{-5}$
$B=0.15$
$n=2.0$

Table 3: Bergstrom-Boyce creep model coefficients.

	Coefficient for model 1	Coefficients for model 2	Coefficients for model 3
A	2.5×10^{-5}	7.15×10^{-5}	2.887
m	4.0	1.0	4.0
C	−1.0	0.5	−1.0
E	0.01	0.01	0.01

Table 4: Darveaux and modified Darveaux creep model coefficients.

$C_{ss}=2.5\times10^{-5}$
α=0.15
n=2.0
ϵ_T=0.025
B=100.0

Table 5: Anand creep model coefficients.

S_1	20.0
A	$2x10^{-5}$
ξ	3.0
m	0.5
h_0	30.0
$\hat{s}$	25.0
n	0.05
a	2.0
S_2	0.0
S_3	0.0
A_1	0.0
A_2	0.0
A_3	0.0
A_4	0.0

Table 6: Hyperelastic materials.

Material	Coefficients for material 1	Coefficients for material 2
Arruda-Boyce	μ=20.0 λ_m=7.0 D=0.0	μ=20.0 λ_m=7.0 D=0.1

Material	Coefficients for material 1	Coefficients for material 2
neo-Hookean	C_{10}=10 D_1=0.1	C_{10}=10 D_1=0.01
Ogden (N=2)	μ_1=16.0 α_1=2.0 μ_2=4.0 α_2=−2.0 D_1=0.0 D_2=0.0	μ_1=16.0 α_1=2.0 μ_2=4.0 α_2=−2.0 D_1=0.1 D_2=0.0
Van der Waals	μ=20.0 λ_m=10.0 α=0.1 β=0.02 D=0.0	μ=20.0 λ_m=10.0 α=0.1 β=0.02 D=0.1

Table 7: Exponential viscous softening coefficients.

η_f^{cr}=0.3
$\bar{\varepsilon}_0^{cr}$=3.0x10^{-4}

Results and discussion

The results agree well with exact analytical or approximate solutions.

Input files

viscnet_c3d8_nh_n2.inp	Uniaxial creep test, C3D8 element and neo-Hookean material.
viscnet_c3d8_nh_n3.inp	Uniaxial creep test, C3D8 element, neo-Hookean material, and three viscoelastic networks.
viscnet_c3d8_nh_n3_md.inp	Uniaxial creep test, C3D8 element, neo-Hookean material, and three viscoelastic networks: strain-hardening power law model, hyperbolic-sine model, and modified Darveaux model.
viscnet_c3d8h_ab_n2.inp	Uniaxial creep test, C3D8H element, and Arruda-Boyce material.
viscnet_c3d8h_ab_n3.inp	Uniaxial creep test, C3D8H element, Arruda-Boyce material, and three viscoelastic networks.
viscnet_c3d8h_ab_n3_md.inp	Uniaxial creep test, C3D8H element, Arruda-Boyce material, and three viscoelastic networks: strain-hardening power law model, hyperbolic-sine model, and modified Darveaux model.
viscnet_c3d8h_ab_bb_n2.inp	Uniaxial creep test, C3D8H element, Arruda-Boyce hyperelastic material, and two viscoelastic networks: strain-hardening power law model, and Bergstrom-Boyce model.
viscnet_c3d8h_ab_bb_n2_user.inp	Uniaxial creep test, C3D8H element, Arruda-Boyce hyperelastic material, and two viscoelastic networks: strain-hardening power law model and

	Bergstrom-Boyce model defined via user subroutine *UCREEPNETWORK* provided in `viscnet_bb_user.f`.
viscnet_bb_user.f	*UCREEPNETWORK* subroutine to define the Bergstrom-Boyce creep model.
viscnet_c3d8h_ab_bb_n3.inp	Uniaxial creep test, C3D8H element, Arruda-Boyce hyperelastic material, and three viscoelastic networks: strain-hardening power law model and two Bergstrom-Boyce models.
viscnet_c3d8h_ab_bb_n3_user.inp	Uniaxial creep test, C3D8H element, Arruda-Boyce hyperelastic material, and three viscoelastic networks: strain-hardening power law model and two Bergstrom-Boyce models defined via user subroutine *UCREEPNETWORK* provided in `viscnet_bb_user.f`.
viscnet_c3d8h_ogden_n2.inp	Uniaxial creep test, C3D8H element and Ogden material.
viscnet_c3d8h_vanwaals_n2.inp	Uniaxial creep test, C3D8H element and Van der Waals material.
viscnet_c3d8r_nh_n2.inp	Uniaxial creep test, C3D8R element and neo-Hookean material.
viscnet_c3d8rh_ab_n2.inp	Uniaxial creep test, C3D8RH element and Arruda-Boyce material.
viscnet_c3d8rh_ogden_n2.inp	Uniaxial creep test, C3D8RH element and Ogden material.
viscnet_c3d8rh_vanwaals_n2.inp	Uniaxial creep test, C3D8RH element and Van der Waals material.
viscnet_cax4_nh_n2.inp	Uniaxial creep test, CAX4 element and neo-Hookean material.
viscnet_cax4h_ab_n2.inp	Uniaxial creep test, CAX4H element and Arruda-Boyce material.
viscnet_cax4h_ogden_n2.inp	Uniaxial creep test, CAX4H element and Ogden material.
viscnet_cax4h_vanwaals_n2.inp	Uniaxial creep test, CAX4H element and Van der Waals material.
viscnet_cax4r_nh_n2.inp	Uniaxial creep test, CAX4R element and neo-Hookean material.
viscnet_cax4rh_ab_n2.inp	Uniaxial creep test, CAX4RH element and Arruda-Boyce material.
viscnet_cax4rh_ogden_n2.inp	Uniaxial creep test, CAX4RH element and Ogden material.
viscnet_cax4h_ogden_bb_n2.inp	Uniaxial creep test, CAX4H element, Ogden hyperelastic material, and two viscoelastic networks: strain-hardening power law model and Bergstrom-Boyce model.
viscnet_cax4rh_vanwaals_n2.inp	Uniaxial creep test, CAX4RH element and Van der Waals material.
viscnet_cax4h_vanwaals_bb_n2.inp	Uniaxial creep test, CAX4H element, Van der Waals hyperelastic material, and two viscoelastic networks: strain-hardening power law model and Bergstrom-Boyce model.
viscnet_cax4h_vanwaals_bb_n3.inp	Uniaxial creep test, CAX4H element, Van der Waals hyperelastic material, and three viscoelastic networks: strain-hardening power law model, Bergstrom-Boyce model, and Darveaux model.
viscnet_cax4h_vanwaals_bb_n3_md.inp	Uniaxial creep test, CAX4H element, Van der Waals hyperelastic material, and three viscoelastic networks: strain-hardening power law model, Bergstrom-Boyce model, and modified Darveaux model.
viscnet_cpe4_nh_n2.inp	Uniaxial creep test, CPE4 element and neo-Hookean material.
viscnet_cpe4h_ab_n2.inp	Uniaxial creep test, CPE4H element and Arruda-Boyce material.
viscnet_cpe4h_ogden_n2.inp	Uniaxial creep test, CPE4H element and Ogden material.
viscnet_cpe4h_vanwaals_n2.inp	Uniaxial creep test, CPE4H element and Van der Waals material.
viscnet_cpe4r_nh_n2.inp	Uniaxial creep test, CPE4R element and neo-Hookean material.
viscnet_cpe4rh_ab_n2.inp	Uniaxial creep test, CPE4RH element and Arruda-Boyce material.
viscnet_cpe4rh_ogden_n2.inp	Uniaxial creep test, CPE4RH element and Ogden material.
viscnet_cpe4rh_ogden_n3.inp	Uniaxial creep test, CPE4RH element, Ogden material, and three viscoelastic networks.

viscnet_cpe4rh_ogden_n3_md.inp	Uniaxial creep test, CPE4RH element, Ogden material, and three viscoelastic networks: strain-hardening power law model, hyperbolic-sine model, and modified Darveaux model.
viscnet_cpe4rh_vanwaals_n2.inp	Uniaxial creep test, CPE4RH element and Van der Waals material.
viscnet_cpe4_nh_bb_n2.inp	Uniaxial creep test, CPE4 element, neo-Hookean hyperelastic material, and two viscoelastic networks: strain-hardening power law model and Bergstrom-Boyce model.
viscnet_c3d8_nh_n4.inp	Uniaxial creep test, C3D8 element, neo-Hookean hyperelastic material, and four viscoelastic networks: strain-hardening power law model, hyperbolic-sine model, modified Darveaux model, and Anand model.
viscnet_c3d8h_ab_n4.inp	Uniaxial creep test, C3D8H element, Arruda-Boyce hyperelastic material, and four viscoelastic networks: strain-hardening power law model, hyperbolic-sine model, modified Darveaux model, and Anand model.
viscnet_cax4h_vanwaals_bb_n4.inp	Uniaxial creep test, CAX4H element, Van der Waals hyperelastic material, and four viscoelastic networks: strain-hardening power law model, Bergstrom-Boyce model, modified Darveaux model, and Anand model.
viscnet_cpe4r_ogden_n4.inp	Uniaxial creep test, CPE4RH element, Ogden material, and four viscoelastic networks: strain-hardening power law model, hyperbolic-sine model, modified Darveaux model, and Anand model.
viscnet_c3d8_nh_expvs_n2.inp	Uniaxial creep test, C3D8 element, and neo-Hookean material with exponential viscous softening in viscoelastic network 1.
viscnet_c3d8_nh_tabvs_n2.inp	Uniaxial creep test, C3D8 element, and neo-Hookean material with tabular viscous softening in viscoelastic network 1.
viscnet_c3d8_nh_expvs_n2_user.inp	Uniaxial creep test, C3D8 element, and neo-Hookean material with exponential viscous softening in viscoelastic network 1 defined via user subroutine *UHYPER* provided in `viscnet_nh_expvs_user.f`.
viscnet_nh_expvs_user.f	User subroutine *UHYPER* to define a neo-Hookean material with exponential viscous softening in viscoelastic network 1.
x_viscnet_c3d8_nh_n2.inp	Explicit dynamic analysis, uniaxial creep test, C3D8 element and neo-Hookean material.
x_viscnet_c3d8_nh_n3.inp	Explicit dynamic analysis, uniaxial creep test, C3D8 element, neo-Hookean material and three viscoelastic networks: strain-hardening power law model, hyperbolic-sine model, and modified Darveaux model.
x_viscnet_c3d8_nh_n3_md.inp	Explicit dynamic analysis, uniaxial creep test, C3D8 element, neo-Hookean material and three viscoelastic networks.
x_viscnet_c3d8_ab_n2.inp	Explicit dynamic analysis, uniaxial creep test, C3D8 element and Arruda-Boyce material.
x_viscnet_c3d8_ab_n3.inp	Explicit dynamic analysis, uniaxial creep test, C3D8 element, Arruda-Boyce material and three viscoelastic networks.
x_viscnet_c3d8_ab_n3_md.inp	Explicit dynamic analysis, uniaxial creep test, C3D8 element, Arruda-Boyce material and three viscoelastic networks: strain-hardening power law model, hyperbolic-sine model, and modified Darveaux model.
x_viscnet_c3d8_ab_n4.inp	Explicit dynamic analysis, uniaxial creep test, C3D8 element and Arruda-Boyce material with four creep networks.
x_viscnet_c3d8_ogden_n2.inp	Explicit dynamic analysis, uniaxial creep test, C3D8 element and Ogden material.
x_viscnet_c3d8_vanwaals_n2.inp	Explicit dynamic analysis, uniaxial creep test, C3D8 element and Van der Waals material.

x_viscnet_c3d8r_nh_n2.inp	Explicit dynamic analysis, uniaxial creep test, C3D8R element and neo-Hookean material.
x_viscnet_c3d8_nh_n4.inp	Explicit dynamic analysis, uniaxial creep test, C3D8R element and neo-Hookean material with four creep networks.
x_viscnet_c3d8r_ab_n2.inp	Explicit dynamic analysis, uniaxial creep test, C3D8R element and Arruda-Boyce material.
x_viscnet_c3d8r_ogden_n2.inp	Explicit dynamic analysis, uniaxial creep test, C3D8R element and Ogden material.
x_viscnet_cpe4r_ogden_n4.inp	Explicit dynamic analysis, uniaxial creep test, CPE4R element and Ogden material with four creep networks.
x_viscnet_c3d8r_vanwaals_n2.inp	Explicit dynamic analysis, uniaxial creep test, C3D8R element and Van der Waals material.
x_viscnet_c3d8r_nh_n1_bb.inp	Explicit dynamic analysis, uniaxial creep test, C3D8R element, neo-Hookean hyperelastic material, and one Bergstrom-Boyce viscoelastic network.
x_viscnet_cax4r_nh_n2.inp	Explicit dynamic analysis, uniaxial creep test, CAX4R element and neo-Hookean material.
x_viscnet_cax4r_ab_n2.inp	Explicit dynamic analysis, uniaxial creep test, CAX4R element and Arruda-Boyce material.
x_viscnet_cax4r_ogden_n2.inp	Explicit dynamic analysis, uniaxial creep test, CAX4R element and Ogden material.
x_viscnet_cax4r_vanwaals_n2.inp	Explicit dynamic analysis, uniaxial creep test, CAX4R element and Van der Waals material.
x_viscnet_cax4r_vanwaals_bb_n4.inp	Explicit dynamic analysis, uniaxial creep test, CAX4R element and Van der Waals material with four creep networks.
x_viscnet_cax4r_vanwaals_bb_n3_md.inp	Explicit dynamic analysis, uniaxial creep test, CAX4R element and Van der Waals material, and three viscoelastic networks: strain-hardening power law model, Bergstrom-Boyce model, and modified Darveaux model.
x_viscnet_cpe4r_nh_n2.inp	Explicit dynamic analysis, uniaxial creep test, CPE4R element and neo-Hookean material.
x_viscnet_cpe4r_ab_n2.inp	Explicit dynamic analysis, uniaxial creep test, CPE4R element and Arruda-Boyce material.
x_viscnet_cpe4r_ogden_n2.inp	Explicit dynamic analysis, uniaxial creep test, CPE4R element and Ogden material.
x_viscnet_cpe4r_ogden_n3.inp	Explicit dynamic analysis, uniaxial creep test, CPE4R element , Ogden material and three viscoelastic networks.
x_viscnet_cpe4r_ogden_n3_md.inp	Explicit dynamic analysis, uniaxial creep test, CPE4R element , Ogden material and three viscoelastic networks: strain-hardening power law model, hyperbolic-sine model, and modified Darveaux model.
x_viscnet_cpe4r_vanwaals_n2.inp	Explicit dynamic analysis, uniaxial creep test, CPE4R element and Van der Waals material.
x_viscnet_c3d8r_nh_expvs_n2.inp	Explicit dynamic analysis, uniaxial creep test, C3D8 element and neo-Hookean material with exponential viscous softening in viscoelastic network 1.
x_viscnet_c3d8r_nh_tabvs_n2.inp	Explicit dynamic analysis, uniaxial creep test, C3D8 element and neo-Hookean material with tabular viscous softening in viscoelastic network 1.

x_viscnet_c3d8r_nh_expvs_n2_user.inp	Explicit dynamic analysis, uniaxial creep test, C3D8 element and neo-Hookean material with exponential viscous softening in viscoelastic network 1 defined via user subroutine *VUHYPER* provided in `x_viscnet_nh_expvs_user.f`.
x_viscnet_nh_expvs_user.f	User subroutine *VUHYPER* to define a neo-Hookean material with exponential viscous softening in viscoelastic network 1.

Transient internal pressure loading of a viscoelastic cylinder

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

CAX4I CAX8R CPE4I CPE8R

Features tested

The automatic incrementation capability provided for integration of time-dependent material models and the use of the viscoelastic material model for a larger number of Prony series terms are tested in this problem. It also demonstrates the use of viscoelastic material models in dynamic analysis.

Problem description

The structure is a solid rocket motor, modeled as a long, hollow viscoelastic cylinder encased in a thin steel shell. The rocket's ignition is simulated by a transient internal pressure load acting at the inner diameter of the viscoelastic cylinder. The transient response of the structure is sought.

Model: The viscoelastic cylinder has an inner radius of 10 mm and an outer radius of 50 mm. The steel case is 0.5 mm thick. We assume plane strain, with no gradient of the solution, in the axial direction. The problem is, therefore, modeled with a single row of axisymmetric, second-order, reduced-integration elements (CAX8R). The viscoelastic material is represented using 20 elements, while the case is modeled with a single element.

Mesh: The mesh is shown in *Figure 1*. The mesh is finer toward the inner diameter of the cylinder, where the stresses are highest.

Material: The extensional relaxation function of the viscoelastic material is defined using a six-term Prony series:

$$E_R(t) = E_0\left(1 - \sum_{i=1}^{6} \overline{g}_i^P \left(1 - e^{-t/\tau_i}\right)\right).$$

The instantaneous modulus, E_0, is 1651.59 MPa; and the six pairs of relative moduli $\overline{g}_i^P$ and time constants τ_i are

i	$\overline{g}_i^P$	τ_i, sec
1	0.1986	0.281×10^{-7}
2	0.1828	0.281×10^{-5}
3	0.1388	0.281×10^{-3}
4	0.2499	0.281×10^{-1}
5	0.1703	0.281×10^{1}
6	0.0593	0.281×10^{3}

This model results in a very low long-term elastic modulus (0.4955 MPa), so the material almost behaves as a viscoelastic fluid. Because the viscoelastic material is incompressible throughout the problem, the relative moduli $\overline{g}_i^P$ and time constants τ_i that constitute the extensional relaxation function can be used directly in the definition of the shear relaxation function. Contrast this with *Viscoelastic rod subjected to constant axial load*, in which the material is slightly compressible, so that the shear modulus and time constant were related to the extensional values through the bulk modulus.

A solution to the present problem is also obtained by modeling the behavior of the viscoelastic cylinder with large-strain linear viscoelastic theory. The relaxation behavior is defined in the same way, but the short-term elastic properties are specified as part of the hyperelastic material definition. The polynomial formulation with $N = 1$ is used, and the constants are C_{10} = 275.247 MPa , C_{01} = 0 (neo-Hookean material) and D_1 = 7. × 10^{-7} MPa^{-1}. These constants are such that the initial Young's modulus and initial Poisson's ratio are equal to E_0 and ν_0, respectively. The steel case is assumed to be linear elastic, with a Young's modulus of 200 GPa and a Poisson's ratio of 0.3.

Loading: The time-dependent pressure loading used in the static analyses is

$$p = 10\left(1 - e^{-23.03t}\right) \text{ MPa.}$$

A plot of this loading as a function of time is shown in *Figure 2*. To highlight inertia effects, the pressure loading in the dynamic analysis is applied 10 times faster:

$$p = 10\left(1 - e^{-230.3t}\right) \text{ MPa.}$$

These pressure histories are specified using user subroutine *DLOAD*.

Analysis

The static analysis is done using the quasi-static procedure with a time period of 0.5 sec. A tolerance on the accuracy of the transient creep solution is specified to enable automatic time incrementation. The accuracy tolerance is set to 7.0 × 10^{-3}, which is the same order of magnitude as the maximum elastic strain.

The dynamic analysis is done using the dynamic procedure with a time period of 0.05 sec. This analysis is done based on nonlinear geometric behavior. A tolerance on the accuracy of the half-increment residual is specified to enable automatic incrementation. The value chosen (1000 N) is one order of magnitude higher than the highest equivalent nodal loads.

Results and discussion

Figure 3 through *Figure 5* depict, respectively, the time histories of the radial stress, hoop stress, and hoop strain in the innermost element for a linear static analysis. The static analysis with the large-strain formulation gives almost identical results.

Figure 6 through *Figure 8* depict, respectively, the time histories of the radial stress, hoop stress, and hoop strain in the innermost element for the nonlinear dynamic analysis.

Input files

viscocylinder_cax8r_linear.inp	Linear static analysis.
viscocylinder_cax8r_linear.f	User subroutine *DLOAD* used in viscocylinder_cax8r_linear.inp.
viscocylinder_cax8r_static.inp	Nonlinear static analysis.
viscocylinder_cax8r_static.f	User subroutine *DLOAD* used in viscocylinder_cax8r_static.inp.

viscocylinder_cax8r_dyn.inp	Nonlinear dynamic analysis.
viscocylinder_cax8r_dyn.f	User subroutine *DLOAD* used in viscocylinder_cax8r_dyn.inp.
viscocylinder_cpe8r.inp	Uses a wedge of plane strain elements (CPE8R) to solve the linear static analysis.
viscocylinder_cpe8r.f	User subroutine *DLOAD* used in viscocylinder_cpe8r.inp.
viscocylinder_cax4i_linear.inp	Linear static analysis using CAX4I.
viscocylinder_cax4i_linear.f	User subroutine *DLOAD* used in viscocylinder_cax4i_linear.inp.
viscocylinder_cax4i_static.inp	Nonlinear static analysis using CAX4I.
viscocylinder_cax4i_static_po.inp	**POST OUTPUT* analysis.
viscocylinder_cax4i_static.f	User subroutine *DLOAD* used in viscocylinder_cax4i_static.inp.
viscocylinder_cax4i_dyn.inp	Nonlinear dynamic analysis using CAX4I.
viscocylinder_cax4i_dyn.f	User subroutine *DLOAD* used in viscocylinder_cax4i_dyn.inp.
viscocylinder_cpe4i.inp	Linear static analysis using CPE4I.
viscocylinder_cpe4i.f	User subroutine *DLOAD* used in viscocylinder_cpe4i.inp.

Figures

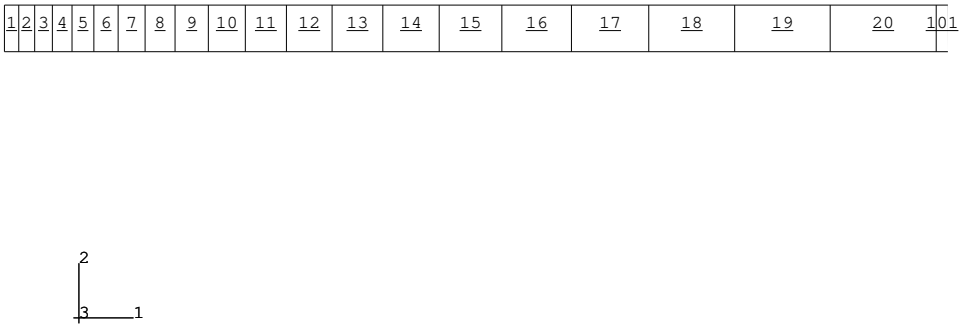


Figure 1: Finite element model of viscoelastic cylinder with elastic case.

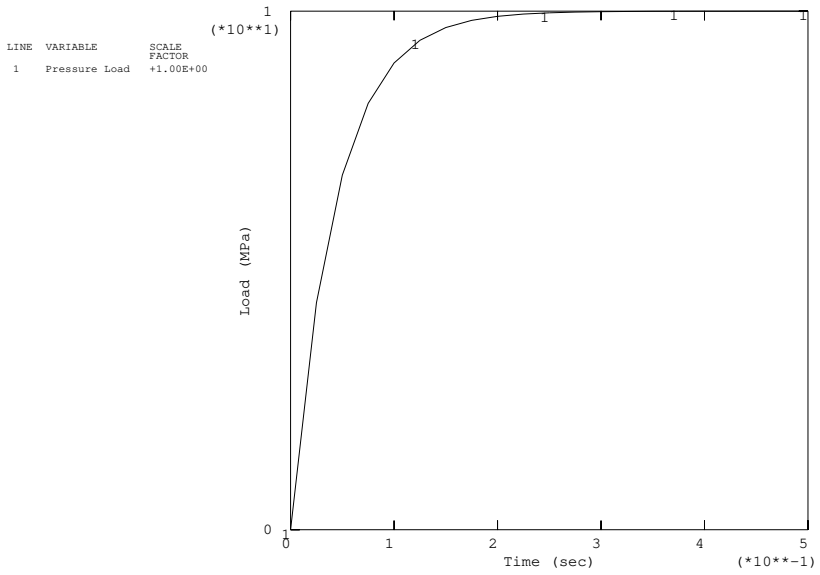


Figure 2: Variation of internal pressure load with time (static analysis).

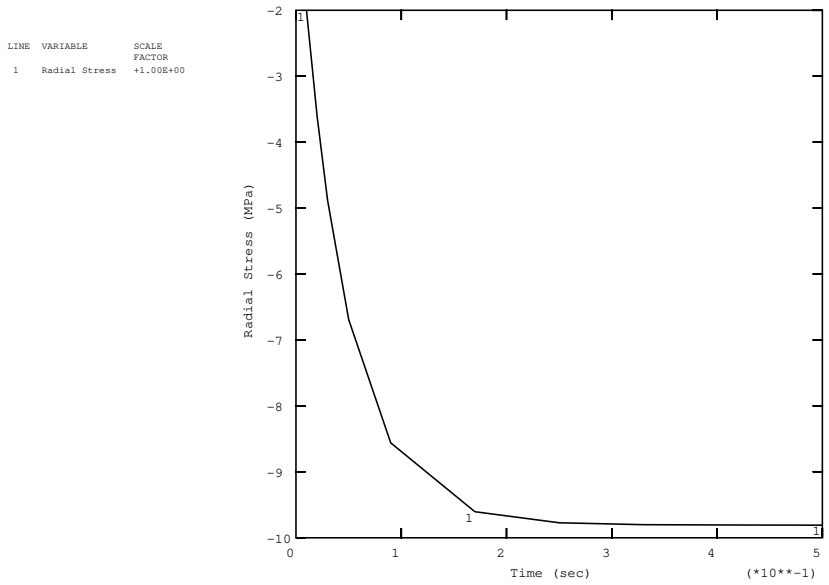


Figure 3: Radial stress at innermost integration point on cylinder (static analysis).

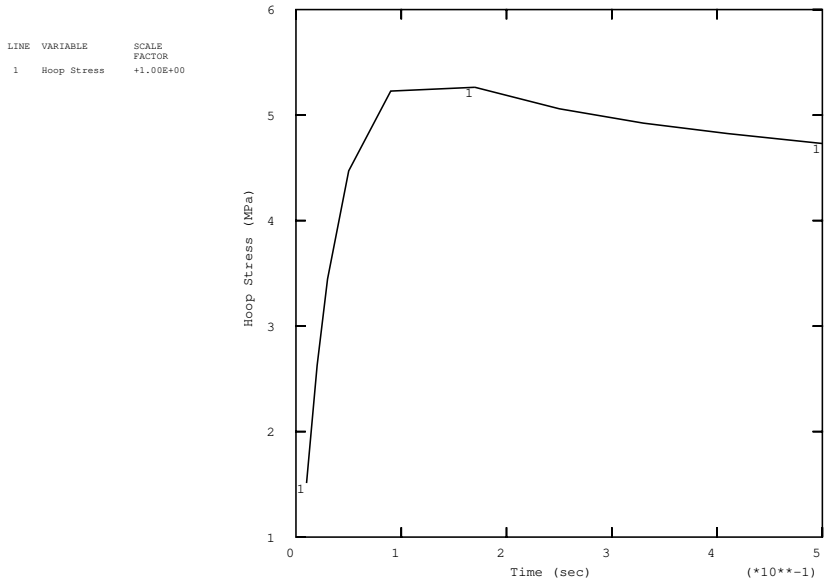


Figure 4: Hoop stress at innermost integration point on cylinder (static analysis).

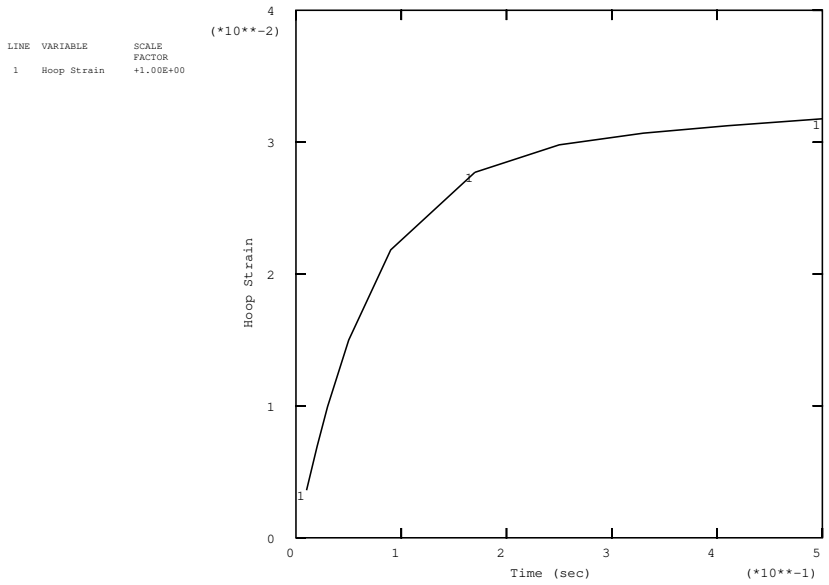


Figure 5: Hoop strain at innermost integration point on cylinder (static analysis).

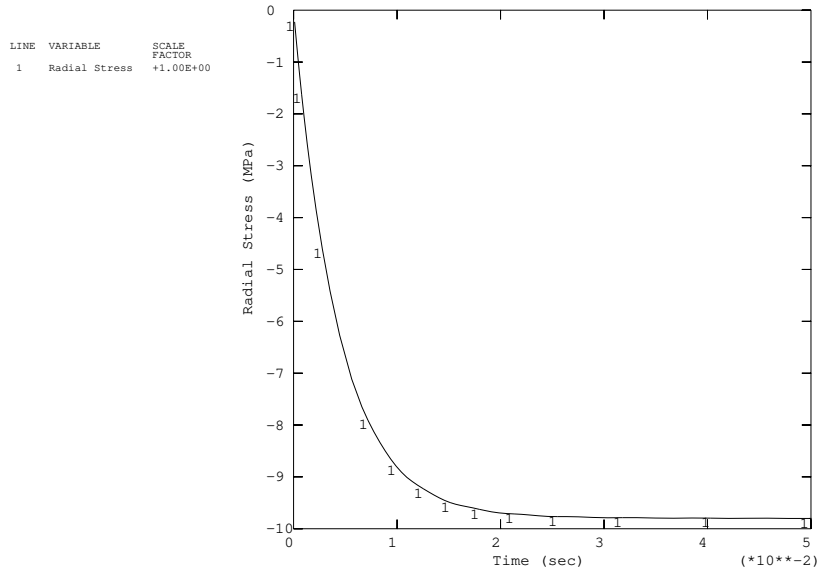


Figure 6: Radial stress at innermost integration point on cylinder (dynamic analysis).

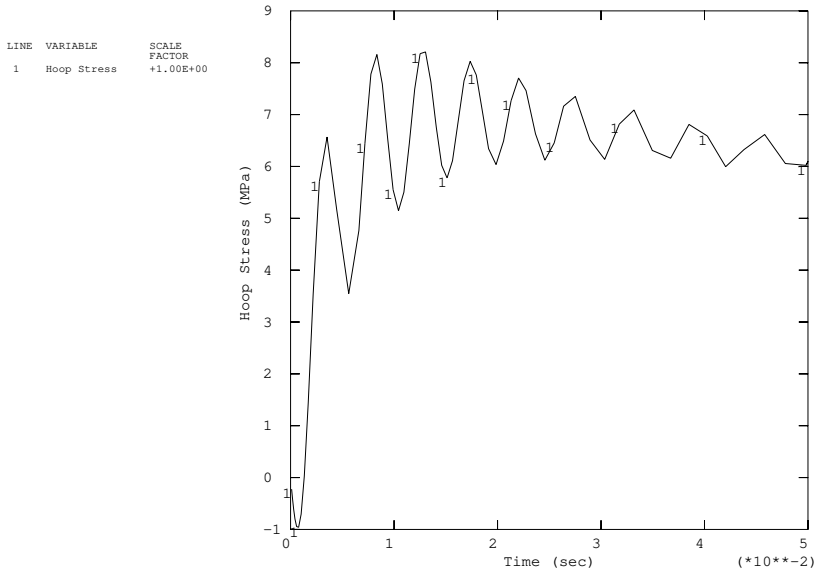


Figure 7: Hoop stress at innermost integration point on cylinder (dynamic analysis).

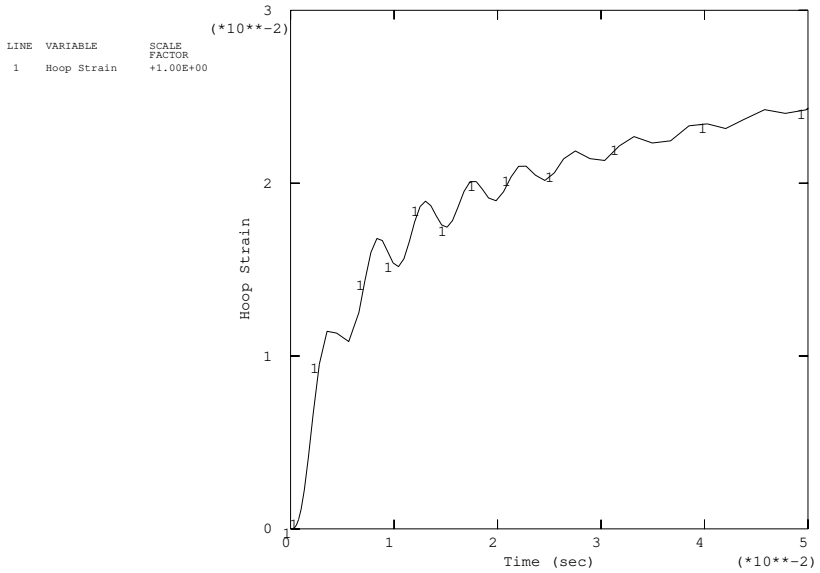


Figure 8: Hoop strain at innermost integration point on cylinder (dynamic analysis).

Rate-independent plasticity

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Mises plasticity with isotropic hardening

Elements tested

C3D8 CPS4 T3D2

Problem description

Material:

Elasticity

Young's modulus, E = 200.0E3

Poisson's ratio, ν = 0.3

Plasticity

Hardening:

Yield stress	Plastic strain
200.	0.0000
220.	0.0009

(The units are not important.)

Results and discussion

The results agree well with exact analytical or approximate solutions.

Input files

Abaqus/Standard input files

mpliho3hut.inp	Uniaxial tension, C3D8 elements.
mpliho3hut_xtrpl.inp	Uniaxial tension, C3D8 elements, linear extrapolation of the yield stress.
mpliho2hut.inp	Uniaxial tension, CPS4 elements.
mpliho1hut.inp	Uniaxial tension, T3D2 elements.
mpliho3gsh.inp	Shear, C3D8 elements.
mpliho2gsh.inp	Shear, CPS4 elements.
mpliho1mcy.inp	Cyclic loading, T3D2 elements.

mpliho3vlp.inp — Linear perturbation steps containing **LOAD CASE*, uniaxial tension, C3D8 elements.

mplihi3hut.inp — Uniaxial tension with nonzero initial condition for $\bar{\varepsilon}^{pl}$, C3D8 elements.

Abaqus/Explicit input files

mpliho3hut_xtrpl_xpl.inp — Uniaxial tension, C3D8 elements, linear extrapolation of the yield stress.

Mises plasticity with linear kinematic hardening

Elements tested

T3D2

Problem description

Material:

Elasticity

Young's modulus, E = 200.0E3

Poisson's ratio, ν = 0.3

Plasticity

Hardening:

Yield stress	Plastic strain
200.	0.0000
220.	0.0009

The linear kinematic hardening model is defined by the slope of the stress-strain data given earlier. (The units are not important.)

Results and discussion

The results agree well with exact analytical or approximate solutions.

Input files

Abaqus/Standard input files

mplkho1mcy.inp — Cyclic loading, T3D2 elements.

mplkhi1hut.inp — Uniaxial tension with nonzero initial condition for α_{11}, load control, T3D2 elements.

Abaqus/Explicit input files

mplkho1mcy_xpl.inp — Cyclic loading, T3D2 elements.

mplkhi1hut_xpl.inp — Uniaxial tension with nonzero initial condition for α_{11}, load control, T3D2 elements.

Mises plasticity with multilinear kinematic hardening

Elements tested

C3D8

Problem description

Material:

Elasticity

Young's modulus, E = 200.0E3

Poisson's ratio, ν = 0.3

Plasticity

Hardening:

Yield stress	Plastic strain	Temperature	Field variable
2000.	0.00	0.	0.
3000.	0.25	0.	0.
5500.	1.0	0.	0.
800.	0.	100.	0.
2000.	0.25	100.	0.
4000.	0.8	100.	0.
400.	0.	0.	200.
800.	0.1	0.	200.
1600.	0.4	0.	200.
200.	0.	100.	200.
500.	0.5	100.	200.
750.	1.	100.	200.

Thermal properties

Coefficient of expansion, α = 1.E-4

Results and discussion

The results agree well with the analytical solution.

Input files

Abaqus/Standard input file

multilinkinhard_fv.inp Uniaxial test, C3D8 element.

Mises plasticity with combined isotropic/kinematic hardening

Elements tested

B21 C3D8 C3D8R CPE4 CPS4 M3D4 SAX1 T3D2

Problem description

Material 1:

Elasticity

Young's modulus, $E = 200.0E3$

Poisson's ratio, $\nu = 0.3$

Plasticity

Initial yield stress: $\sigma_0 = 200.0$

Isotropic hardening parameter, $Q_\infty = 2000$

Isotropic hardening parameter, $b = 0.25$

Kinematic hardening parameter, $C = 2.222 \times 10^4$

Kinematic hardening parameter, $\gamma = 34.65$

The parameters given above are used to generate data for some of the input files that use tabular data. (The units are not important.)

Material 2:

Elasticity

Young's modulus, $E = 200.0E3$

Poisson's ratio, $\nu = 0.3$

Plasticity

Initial yield stress: $\sigma_0 = 200.0$

Kinematic hardening parameter, $C = 2.222 \times 10^4$

Kinematic hardening parameter, $\gamma = 0.0$

The parameters given above are used to generate data for some of the input files that use tabular data. (The units are not important.)

Material 3:

Elasticity

Young's modulus, $E = 200.0E3$

Poisson's ratio, $\nu = 0.3$

Plasticity

Initial yield stress: σ_0 = 200.0

Isotropic hardening parameter, Q_∞ = 0.0

Isotropic hardening parameter, b = 0.0

Kinematic hardening parameter, $C = 2.222 \times 10^4$

Kinematic hardening parameter, γ = 34.65

(The units are not important.)

Material 4:

Elasticity

Young's modulus, E = 200.0E3

Poisson's ratio, ν = 0.3

Plasticity

Initial yield stress: σ_0 = 200.0

Isotropic hardening parameter, Q_∞ = 2000

Isotropic hardening parameter, b = 0.25

Kinematic hardening parameter, $C_1 = 1.111 \times 10^4$

Kinematic hardening parameter, γ_1 = 34.65

Kinematic hardening parameter, $C_2 = 5.555 \times 10^3$

Kinematic hardening parameter, γ_2 = 34.65

Kinematic hardening parameter, $C_3 = 5.555 \times 10^3$

Kinematic hardening parameter, γ_3 = 0.0

(The units are not important.)

Material 5:

Elasticity

Young's modulus, E = 200.0E3

Poisson's ratio, ν = 0.3

Plasticity

Initial yield stress: σ_0 = 200.0

Isotropic hardening parameter, Q_∞ = 0.0

Isotropic hardening parameter, $b = 0.0$

Kinematic hardening parameter, $C_1 = 1.111 \times 10^4$

Kinematic hardening parameter, $\gamma_1 = 34.65$

Kinematic hardening parameter, $C_2 = 5.555 \times 10^3$

Kinematic hardening parameter, $\gamma_2 = 34.65$

Kinematic hardening parameter, $C_3 = 5.555 \times 10^3$

Kinematic hardening parameter, $\gamma_3 = 0.0$

(The units are not important.)

Material 6:

Elasticity

Young's modulus, $E = 200.0E3$

Poisson's ratio, $\nu = 0.3$

Plasticity

Hardening:

Yield stress	Plastic strain
200.	0.0000
209.975	0.02
214.9325	0.03

Isotropic hardening parameter, $b = 0.25$

Kinematic hardening parameter, $C = 2.222 \times 10^4$

Kinematic hardening parameter, $\gamma = 34.65$

(The units are not important.)

Results and discussion

The results agree well with exact analytical or approximate solutions.

Input files

Abaqus/Standard input files

mplchi3nt1mb.inp — Uniaxial tension, nonzero initial conditions for α_{11}, $\bar{\varepsilon}^{pl}$, and σ_{11}; displacement control; M3D4 elements with rebar; number backstresses = 3.

Material 1:

mplchb2hut.inp — Uniaxial tension with temperature and field variable dependence, displacement control, SAX1 elements.

mplchb3hut.inp — Uniaxial tension with temperature-dependent γ, displacement control, C3D8R elements.

mplcho1hut.inp — Uniaxial tension, tabulated data, load control, B21 elements.

mplcho1hutmb.inp — Uniaxial tension, tabulated data, load control, B21 elements, number backstresses = 3.

mplcho3nt1.inp — Uniaxial tension, load control, C3D8 elements.

mplchi3nt1.inp — Uniaxial tension, nonzero initial conditions for α_{11}, $\bar{\varepsilon}^{pl}$, and σ_{11}; displacement control; M3D4 elements with rebar.

mplchi2hut.inp — Uniaxial tension with orientation and nonzero initial conditions for α_{11} and $\bar{\varepsilon}^{pl}$, displacement control, CPE4 elements.

Material 2:

mplcho1mcy.inp — Cyclic loading, no isotropic hardening, displacement control, T3D2 elements.

mplcho1mcymb.inp — Cyclic loading, no isotropic hardening, displacement control, T3D2 elements, number backstresses = 3.

Material 3:

mplcho2gsh.inp — Simple shear including perturbation step, CPS4 elements.

Material 4:

mplchb2hutmb.inp — Uniaxial tension with temperature and field variable dependence, displacement control, SAX1 elements, number backstresses = 3.

mplchi2hutmb.inp — Uniaxial tension with orientation and nonzero initial conditions for α_{11} and $\bar{\varepsilon}^{pl}$, displacement control, CPE4 elements, number backstresses = 3.

mplcho3nt1mb.inp — Uniaxial tension, load control, C3D8 elements, number backstresses = 3.

Material 5:

mplcho2gshmb.inp — Simple shear including perturbation step, CPS4 elements, number backstresses = 3.

Material 6:

mplcho3nt1_xtrpl.inp — Uniaxial tension, load control, C3D8 elements, linear extrapolation of the yield stress.

Abaqus/Explicit input files

mplchi3nt1mb_xpl.inp — Uniaxial tension, nonzero initial conditions for α_{11}, $\bar{\varepsilon}^{pl}$, and σ_{11}; displacement control; M3D4R elements with rebar; number backstresses = 3.

Material 1:

mplchb2hut_xpl.inp — Uniaxial tension with temperature and field variable dependence, displacement control, SAX1 elements.

mplchb3hut_xpl.inp — Uniaxial tension with temperature-dependent γ, displacement control, C3D8R elements.

mplcho1hut_xpl.inp — Uniaxial tension, tabulated data, load control, B21 elements.

mplcho1hutmb_xpl.inp — Uniaxial tension, tabulated data, load control, B21 elements, number backstresses = 3.

mplcho3nt1_xpl.inp — Uniaxial tension, load control, C3D8R elements.

mplchi3nt1_xpl.inp — Uniaxial tension, nonzero initial conditions for α_{11}, $\bar{\varepsilon}^{pl}$, and σ_{11}; displacement control; M3D4R elements with rebar.

mplchi2hut_xpl.inp — Uniaxial tension with orientation and nonzero initial conditions for α_{11} and $\bar{\varepsilon}^{pl}$, displacement control, CPE4R elements.

Material 2:

mplcho1mcy_xpl.inp — Cyclic loading, no isotropic hardening, displacement control, T3D2 elements.

mplcho1mcymb_xpl.inp — Cyclic loading, no isotropic hardening, displacement control, T3D2 elements, number backstresses = 3.

Material 3:

mplcho2gsh_xpl.inp — Simple shear including perturbation step, CPS4R elements.

Material 4:

mplchb2hutmb_xpl.inp — Uniaxial tension with temperature and field variable dependence, displacement control, SAX1 elements, number backstresses = 3.

mplchi2hutmb_xpl.inp — Uniaxial tension with orientation and nonzero initial conditions for α_{11} and $\bar{\varepsilon}^{pl}$, displacement control, CPE4R elements, number backstresses = 3.

mplcho3nt1mb_xpl.inp — Uniaxial tension, load control, C3D8R elements, number backstresses = 3.

Material 5:

mplcho2gshmb_xpl.inp — Simple shear including perturbation step, CPS4R elements, number backstresses = 3.

Material 6:

mplcho3nt1_xtrpl_xpl.inp — Uniaxial tension, load control, C3D8 elements, linear extrapolation of the yield stress.

Adiabatic Mises plasticity

Elements tested

C3D8 CPS4 T3D2

Problem description

Material:

Elasticity

Young's modulus, E = 30.0E6

Poisson's ratio, ν = 0.3

Plasticity

Hardening:

Yield stress	Plastic strain	Temperature
30.0E3	0.000	0.0

Yield stress	Plastic strain	Temperature
50.0E3	0.200	0.0
50.0E3	2.000	0.0
3.0E3	0.000	100.0
5.0E3	0.200	100.0
5.0E3	2.000	100.0

Other properties

Density, μ = 1000.0

Specific heat, c = 0.4

Inelastic heat fraction = 0.5

(The units are not important.)

Results and discussion

The results agree well with exact analytical or approximate solutions.

Input files

mhliho3hut.inp	Uniaxial tension, C3D8 elements.
mhliho1hut.inp	Uniaxial tension, T3D2 elements.
mhliho3gsh.inp	Shear, C3D8 elements.
mhliho2gsh.inp	Shear, CPS4 elements.
mhliho3ltr.inp	Triaxial, C3D8 elements.
mhliht3hut.inp	Uniaxial tension, C3D8 elements.
mhliht3xmx.inp	Multiaxial, C3D8 elements.

Hill plasticity

Elements tested

C3D8

Problem description

Material:

Elasticity

Young's modulus, E = 200.0E3

Poisson's ratio, ν = 0.3

Plasticity

Hardening:

Yield stress	Plastic strain
200.	0.0000
220.	0.0009
220.	0.0029

Anisotropic yield ratios: 1.5, 1.2, 1.0, 1.0, 1.0, 1.0

(The units are not important.)

Results and discussion

The results agree well with exact analytical or approximate solutions.

Input files

mppiho3nt1.inp	Uniaxial tension in direction 1, C3D8 elements.
mppiho3ot2.inp	Uniaxial tension in direction 2, C3D8 elements.
mppiho3pt3.inp	Uniaxial tension in direction 3, C3D8 elements.
mppiho3vlp.inp	Linear perturbation steps containing **LOAD CASE*, uniaxial tension in direction 1, C3D8 elements.

Deformation plasticity

Elements tested

C3D8 CPS4 T3D2

Problem description

Material:

Elasticity

Young's modulus, E = 200.0E3

Poisson's ratio, ν = 0.3

Plasticity

Yield stress, σ^0 = 200.0

Exponent, n = 21.315

Yield offset, α = 0.11802

(The units are not important.)

Results and discussion

The results agree well with exact analytical or approximate solutions.

Input files

mdfooo3hut.inp	Uniaxial tension, C3D8 elements.
mdfooo3huti.inp	Uniaxial tension with initial stress, C3D8 elements.

mdfooo2hut.inp	Uniaxial tension, CPS4 elements.
mdfooo2huti.inp	Uniaxial tension with initial stress, CPS4 elements.
mdfooo1hut.inp	Uniaxial tension, T3D2 elements.
mdfooo1huti.inp	Uniaxial tension with initial stress, T3D2 elements.

Drucker-Prager plasticity with linear elasticity

Elements tested

C3D8 C3D8R CAX4 CPE4 CPS4

Problem description

Material:

Elasticity

Young's modulus, E = 300.0E3

Poisson's ratio, ν = 0.3

Plasticity

Angle of friction, β = 40.0

Dilation angle, ψ = 40.0

Third invariant ratio, K = 0.78 (when included; otherwise, 1.0)

Hardening curve:

Yield stress	Plastic strain
6.0E3	0.000000
9.0E3	0.020000
11.0E3	0.063333
12.0E3	0.110000
12.0E3	1.000000

(The units are not important.)

The hyperbolic and exponent forms of the yield criteria are verified by using parameters that reduce them into equivalent linear forms. Reducing the hyperbolic yield function into a linear form requires that $p_t|_0 = d/\tan\beta$. Reducing the exponent yield function into a linear form requires that b = 1.0 and that $a = (\tan\beta)^{-1}$.

Results and discussion

Most tests in this section are set up as cases of the homogeneous deformation of a single element of unit dimensions. Consequently, the results are identical for all integration points within the element. To test certain conditions, however, it is necessary to set up inhomogeneous deformation problems. In each case, the constitutive path is integrated with 20 increments of fixed size.

Input files

Shear criterion: linear Drucker-Prager

mdeooo3euc.inp	Uniaxial compression, C3D8 elements.
mdeooo2euc.inp	Uniaxial compression, CPS4 elements.
mdeooo3ctc.inp	Triaxial compression, CAX4 elements.
mdeooo3dte.inp	Triaxial extension, CAX4 elements.
mdekoo3dte.inp	$K = 0.78$, triaxial extension, CAX4 elements.
mdeooo3gsh.inp	Shear, C3D8 elements.
mdeooo2gsh.inp	Shear, CPS4 elements.
mdekoo3gsh.inp	$K = 0.78$, shear, C3D8 elements.
mdekoo2gsh.inp	$K = 0.78$; shear, CPS4 elements.
mdeooo3hut.inp	Uniaxial tension, C3D8 elements.
mdeooo2hut.inp	Uniaxial tension, CPS4 elements.
mdekoo3hut.inp	$K = 0.78$, uniaxial tension, C3D8 elements.
mdekoo2hut.inp	$K = 0.78$, uniaxial tension, CPS4 elements.
mdekot3hut.inp	$K = 0.78$, uniaxial tension with temperature dependence, C3D8R elements.
mdeooo3jht.inp	Hydrostatic tension, C3D8 elements.
mdeooo3ltr.inp	Triaxial stress, CPE4 elements (inhomogeneous).
mdeooo2ltr.inp	Triaxial stress, CPS4 elements (inhomogeneous).
mdekoo3ltr.inp	$K = 0.78$, triaxial stress, CPE4 elements (inhomogeneous).
mdekoo2ltr.inp	$K = 0.78$, triaxial stress, CPS4 elements (inhomogeneous).
mdeoot3euc.inp	Uniaxial compression with temperature dependence, C3D8 elements.
mdeooo3vlp.inp	Linear perturbation uniaxial compression, C3D8 elements.
mdedos3euc.inp	Uniaxial compression with rate dependence, C3D8 elements.
mdeooi3euc.inp	Uniaxial compression with nonzero initial condition for $\bar{\varepsilon}^{pl}$, C3D8 elements.
mdedoo2euc.inp	Uniaxial tension, perfect plasticity, CPS4 elements.

Shear criterion: exponent

mdeeoo3jht.inp	Hydrostatic tension, C3D8 elements.
mdeeoo3ltr.inp	Triaxial stress, CPE4 elements (inhomogeneous).
mdeeoo3dte.inp	Triaxial extension, CAX4 elements.
mdeeoo3hut.inp	Uniaxial tension, C3D8 elements.
mdeeoo2hut.inp	Uniaxial tension, CPS4 elements.
mdeeoo3gsh.inp	Shear, C3D8 elements.
mdeeoo2gsh.inp	Shear, CPS4 elements.
mdeeoo3ctc.inp	Triaxial compression, CAX4 elements.
mdeeoo3euc.inp	Uniaxial compression, C3D8 elements.
mdeeoo2euc.inp	Uniaxial compression, CPS4 elements.
mdeeos3euc.inp	Uniaxial compression with rate dependence, C3D8 elements.

mdeeot3euc.inp	Uniaxial compression with temperature dependence, C3D8 elements.
mdeeoo3vlp.inp	Linear perturbation uniaxial compression, C3D8 elements.

Abaqus/Standard input files

mdeeoo3jht_xpl.inp	Hydrostatic tension, C3D8R elements.
mdeeoo3ltr_xpl.inp	Triaxial stress, CPE4R elements (inhomogeneous).
mdeeoo3dte_xpl.inp	Triaxial extension, CAX4R elements.
mdeeoo3hut_xpl.inp	Uniaxial tension, C3D8R elements.
mdeeoo2hut_xpl.inp	Uniaxial tension, CPS4R elements.
mdeeoo3gsh_xpl.inp	Shear, C3D8R elements.
mdeeoo2gsh_xpl.inp	Shear, CPS4R elements.
mdeeoo3ctc_xpl.inp	Triaxial compression, CAX4R elements.
mdeeoo3euc_xpl.inp	Uniaxial compression, C3D8R elements.
mdeeoo2euc_xpl.inp	Uniaxial compression, CPS4R elements.
mdeeos3euc_xpl.inp	Uniaxial compression with rate dependence, C3D8R elements.
mdeeot3euc_xpl.inp	Uniaxial compression with temperature dependence, C3D8R elements.

Abaqus/Explicit input files

Shear criterion: exponent with test data

mdeeto3jht.inp	Hydrostatic tension, C3D8 elements.
mdeeto3ltr.inp	Triaxial stress, CPE4 elements (inhomogeneous).
mdeeto3dte.inp	Triaxial extension, CAX4 elements.
mdeeto3hut.inp	Uniaxial tension, C3D8 elements.
mdeeto2hut.inp	Uniaxial tension, CPS4 elements.
mdeeto3gsh.inp	Shear, C3D8 elements.
mdeeto2gsh.inp	Shear, CPS4 elements.
mdeeto3ctc.inp	Triaxial compression, CAX4 elements.
mdeeto3euc.inp	Uniaxial compression, C3D8 elements.
mdeeto2euc.inp	Uniaxial compression, CPS4 elements.
mdeets3euc.inp	Uniaxial compression with rate dependence, C3D8 elements.
mdeeto3vlp.inp	Linear perturbation uniaxial compression, C3D8 elements.
mdeeto3jht_xpl.inp	Hydrostatic tension, C3D8R elements.
mdeeto3ltr_xpl.inp	Triaxial stress, CPE4R elements (inhomogeneous).
mdeeto3dte_xpl.inp	Triaxial extension, CAX4R elements.
mdeeto3hut_xpl.inp	Uniaxial tension, C3D8R elements.
mdeeto2hut_xpl.inp	Uniaxial tension, CPS4R elements.
mdeeto3gsh_xpl.inp	Shear, C3D8R elements.
mdeeto2gsh_xpl.inp	Shear, CPS4R elements.
mdeeto3ctc_xpl.inp	Triaxial compression, CAX4R elements.
mdeeto3euc_xpl.inp	Uniaxial compression, C3D8R elements.
mdeeto2euc_xpl.inp	Uniaxial compression, CPS4R elements.
mdeets3euc_xpl.inp	Uniaxial compression with rate dependence, C3D8R elements.

Abaqus/Standard input filesAbaqus/Explicit input files

Shear criterion: hyperbolic

mdehoo3jht.inp	Hydrostatic tension, C3D8 elements.
mdehoo3ltr.inp	Triaxial stress, CPE4 elements (inhomogeneous).
mdehoo3dte.inp	Triaxial extension, CAX4 elements.
mdehoo3hut.inp	Uniaxial tension, C3D8 elements.
mdehoo2hut.inp	Uniaxial tension, CPS4 elements.
mdehoo3gsh.inp	Shear, C3D8 elements.
mdehoo2gsh.inp	Shear, CPS4 elements.
mdehoo3ctc.inp	Triaxial compression, CAX4 elements.
mdehoo3euc.inp	Uniaxial compression, C3D8 elements.
mdehoo2euc.inp	Uniaxial compression, CPS4 elements.
mdehos3euc.inp	Uniaxial compression with rate dependence, C3D8 elements.
mdehot3euc.inp	Uniaxial compression with temperature dependence, C3D8 elements.
mdehoo3vlp.inp	Linear perturbation uniaxial compression, C3D8 elements.
mdehoo3jht_xpl.inp	Hydrostatic tension, C3D8R elements.
mdehoo3ltr_xpl.inp	Triaxial stress, CPE4R elements (inhomogeneous).
mdehoo3dte_xpl.inp	Triaxial extension, CAX4R elements.
mdehoo3hut_xpl.inp	Uniaxial tension, C3D8R elements.
mdehoo2hut_xpl.inp	Uniaxial tension, CPS4R elements.
mdehoo3gsh_xpl.inp	Shear, C3D8R elements.
mdehoo2gsh_xpl.inp	Shear, CPS4R elements.
mdehoo3ctc_xpl.inp	Triaxial compression, CAX4R elements.
mdehoo3euc_xpl.inp	Uniaxial compression, C3D8R elements.
mdehoo2euc_xpl.inp	Uniaxial compression, CPS4R elements.
mdehos3euc_xpl.inp	Uniaxial compression with rate dependence, C3D8R elements.
mdehot3euc_xpl.inp	Uniaxial compression with temperature dependence, C3D8R elements.

Abaqus/Standard input filesAbaqus/Explicit input files

Transferring results between Abaqus/Standard and Abaqus/Explicit

sx_s_druckerprager.inp	Base problem for carrying out import from Abaqus/Standard to Abaqus/Explicit, C3D8R elements, uniaxial tension.
sx_x_druckerprager_y_y.inp	Explicit dynamic continuation of sx_s_druckerprager.inp with both the reference configuration and the state imported, C3D8R elements, uniaxial tension.
sx_x_druckerprager_n_y.inp	Explicit dynamic continuation of sx_s_druckerprager.inp with only the state imported, C3D8R elements, uniaxial tension.
sx_x_druckerprager_n_n.inp	Explicit dynamic continuation of sx_s_druckerprager.inp without importing the state or the reference configuration, C3D8R elements, uniaxial tension.
xs_s_druckerprager_y_y.inp	Import into Abaqus/Standard from sx_x_druckerprager_y_y.inp with both the reference configuration and the state imported, C3D8R elements, uniaxial tension.

xs_s_druckerprager_n_y.inp	Import into Abaqus/Standard from sx_x_druckerprager_n_y.inp with only the state imported, C3D8R elements, uniaxial tension.
xs_s_druckerprager_n_n.inp	Import into Abaqus/Standard from sx_x_druckerprager_n_n.inp without importing the state or the reference configuration, C3D8R elements, uniaxial tension.

Drucker-Prager plasticity with porous elasticity

Elements tested

CAX4

Problem description

Material:

Elasticity

Logarithmic bulk modulus, $\kappa = 1.49$

Poisson's ratio, $\nu = 0.1$

Plasticity

Angle of friction, $\beta = 10.0$

Dilation angle, $\psi = 10.0$

Hardening curve:

Yield stress	Plastic strain
100.0	0.0
500.0	0.5

Initial conditions

Initial void ratio, $e_0 = 4.1$

The hyperbolic and exponent forms of the yield criteria are verified by using parameters that reduce them into equivalent linear forms. Reducing the hyperbolic yield function into a linear form requires that $p_t|_0 = d/\tan\beta$. Reducing the exponent yield function into a linear form requires that $b = 1.0$ and that $a = (\tan\beta)^{-1}$.

(The units are not important.)

Results and discussion

The tests in this section are set up as cases of homogeneous deformation of a single element of unit dimensions. Consequently, the results are identical for all integration points within the element. In each case, the constitutive path is integrated with 20 increments of fixed size.

Input files

Shear criterion: linear Drucker-Prager

mdpdoo3bus.inp	Uniaxial strain, CAX4 elements.
mdpdoo3ctc.inp	Triaxial compression, CAX4 elements.

Shear criterion: exponent

mdpeoo3bus.inp	Uniaxial strain, CAX4 elements.
mdpeoo3ctc.inp	Triaxial compression, CAX4 elements.

Shear criterion: exponent with test data

mdpeto3bus.inp	Uniaxial strain, CAX4 elements.
mdpeto3ctc.inp	Triaxial compression, CAX4 elements.

Shear criterion: hyperbolic

mdphoo3bus.inp	Uniaxial strain, CAX4 elements.
mdphoo3ctc.inp	Triaxial compression, CAX4 elements.

Cap plasticity

Elements tested

C3D8R CAX4 CPE4

Problem description

Material: In the tests described in this section, the following data for linear elasticity, cap plasticity I, cap hardening I, and $K = 1.0$ are used unless otherwise specified. With this data, the elastic shear modulus is 5000.0 and the bulk modulus is 10000.0. First yield in pure shear occurs at S12 = 100.0, first yield in pure hydrostatic compression occurs at PRESS = 270.0, first yield in pure hydrostatic tension occurs at PRESS = 300.0, and first yield with PRESS = p_a occurs at PRESS = 120.0 and S12 = 125.0. C3D8 elements are used unless otherwise specified.

Linear elasticity (used in nearly all tests)

Young's modulus, $E = 12857.1429$

Poisson's ratio, $\nu = 0.28571429$ (= 1/7)

Cap plasticity I (used in nearly all tests)

Cohesion, $d = 173.20508$ (= $100\sqrt{3}$)

Slope of Drucker-Prager failure surface, $\beta = 30.0$

Cap ellipticity, $R = 0.61858957$

Initial volumetric plastic strain, $\varepsilon_{vol}^{pl}(0) = 0.027$

Transition parameter, α = 0.69258232

Third invariant factor, K = 1.0 or 0.8, depending on the test.

Cap hardening I (used in nearly all tests)

Position of the yield surface in pure hydrostatic compression, p_b

Volumetric compressive plastic strain, ε_{vol}^{pl}

p_b	ε_{vol}^{pl}
213.0	0.00
222.0	0.01
242.0	0.02
282.0	0.03
362.0	0.04
522.0	0.05
842.0	0.06
1482.0	0.07
2762.0	0.08

Cap plasticity II

d = 0.2286E6

β = 85.0

R = 0.0875

$\varepsilon_{vol}^{pl}(0)$ = 1.22

α = 0.07877

K = 1.0

Cap hardening II

Position of the yield surface in pure hydrostatic compression, p_b

Volumetric compressive plastic strain, ε_{vol}^{pl}

p_b	ε_{vol}^{pl}
0.03E6	0.0
0.20E6	1.22
2.00E6	2.44
2.00E7	3.66

Porous elasticity I

Logarithmic bulk modulus, κ = 20.0

Poisson's ratio, ν = 0.28571429

Tensile strength limit, p_t = 1.0E5

Porous elasticity II

κ = 0.09

ν = 0.0

p_t = 0.02E6

Initial conditions

Initial void ratio, e_0 = 1.0

Results and discussion

The results agree well with exact analytical or approximate solutions.

Input files

mcaooo3mcy.inp	Hydrostatic cyclic test, displacement control. The following six steps are executed: **1.** Load, yielding in hydrostatic compression **2.** Unload, still in compression **3.** Reload, yielding in compression **4.** Unload in compression and load, yielding in tension **5.** Unload in tension and load, yielding in compression **6.** Unload
mcaooo3euc.inp	Uniaxial compressive stress test; displacement control. Step 2 reverses the displacement causing yielding in tension.
mcaooo3gsh.inp	Shear test; load control (S22 = −S11); overlaid soft linear element. There will be some hydrostatic stress due to transverse restraint.
mcaooo3ucs.inp	Cyclical shear test; displacement control; S12 dominant.
mcaoot3ctc.inp	Hydrostatic compression to $p = p_a$, then pure shear; displacement control; temperature dependence. Yielding should be volume preserving.
mcakoo3gsh.inp	Shear test; load control; two primary elements and two overlaid soft elements. One set loaded with principal stresses $\propto$ (1, 1, −2), the other with $\propto$ (−1, −1, 2). The ratio of yield stresses should be K = 0.8.

mcaoob3bus.inp	Uniaxial compressive strain (odometer) test; CPE4 element; load control; with temperature and field variable dependence of the **CAP PLASTICITY* and **CAP HARDENING* data. The temperatures and field variables are specified to give **CAP PLASTICITY* and **CAP HARDENING* data exactly the same as cap plasticity I and cap hardening I data.
mcaooo3bus.inp	Uniaxial compressive strain (odometer) test; load control; NLGEOM and porous elasticity I. The tangent bulk and shear moduli of porous elasticity I differ from that of the linear elasticity by about 1% over the strain range of the test.
mcaooo3ctc.inp	Triaxial test. Hydrostatic loading to $p = p_a$, then increase S11 only.
mcaooo3vlp.inp	Uniaxial compressive strain (odometer) test; load control; the nonlinear analysis is split into two steps, each of which is preceded by a linear perturbation step. The results of the nonlinear steps should correspond to those of mca0003bus.inp. The results of the two linear perturbation steps (**STATIC*) should be identical because small displacements are assumed and the elasticity is linear.
mcakoo3ltr.inp	A displacement pattern designed to produce different stress states at the 8 Gauss points but dominated by shear. The aim is to test the robustness of the Newton loops, so very large strain increments are taken. Displacement control. $K = 0.8$.
mcaooo3ltr.inp	Another test of the robustness of the algorithm. CAX4 element, porous elasticity II, cap plasticity II, and cap hardening II is used.
mcaooo3xmx.inp	Tests adjustment of the initial position of the cap. Two C3D8R elements with different initial stress states. The initial stress in element 1 will cause an adjustment that will make the stress point lie on the cap yield surface. The initial stress in element 2 will cause an adjustment that will make the stress point lie on the transition yield surface.

Clay plasticity with porous elasticity

Elements tested

C3D8 CAX8R

Problem description

Material:

Porous elasticity

Logarithmic bulk modulus, $\kappa = 0.026$

Poisson's ratio, $\nu = 0.3$

Plasticity

Logarithmic plastic bulk modulus, $\lambda = 0.174$

Critical state slope, $M = 1.0$

Initial yield surface size, $\alpha_0 = 58.3$

(except in tests mcl*xxxx*ahc.inp where we use $\alpha_0 = 130.9$ or $e_1 = 1.904$)

Cap parameter, $\beta = 0.5$ (when included; otherwise, 1.0)

Third invariant ratio, $K = 0.78$ (when included; otherwise, 1.0)

Initial conditions

Initial void ratio, $e_0 = 1.08$

(The units are not important.)

Results and discussion

The results agree well with exact analytical or approximate solutions.

Input files

mclooo3ahc.inp	Hydrostatic compression, C3D8 elements.
mcloio3ahc.inp	Hydrostatic compression with intercept option, C3D8 elements.
mclooo3ctc.inp	Triaxial compression, CAX8R elements.
mclott3ctc.inp	Triaxial compression, temperature dependence, CAX8R elements.
mclobo3ctc.inp	$\beta = 0.5$, triaxial compression, CAX8R elements.
mclooo3dte.inp	Triaxial extension, CAX8R elements.
mclkoo3dte.inp	$K = 0.78$, triaxial extension, CAX8R elements.
mclktd3dte.inp	$K = 0.78$, triaxial extension, field variable dependence, CAX8R elements.
mclkbo3dte.inp	$\beta = 0.5$, $K = 0.78$, triaxial extension, CAX8R elements.
mcloto3euc.inp	Uniaxial compression, CAX8R elements.
mclooo3gsh.inp	Shear, C3D8 elements.
mcloto3gsh.inp	Shear, tabulated hardening, C3D8 elements.
mclooo3vlp.inp	Linear perturbation hydrostatic compression, C3D8 elements.

Crushable foam plasticity

Elements tested

C3D8 CPE4

Problem description

Material:

Elasticity

Young's modulus, E = 3.0E6

Poisson's ratio, ν = 0.2

Plasticity

Initial yield stress in hydrostatic compression, p_0 = 2.0E5

Strength in hydrostatic tension, p_t = 2.0E4

Initial yield stress in uniaxial compression, σ_0 = 2.2E5

Yield stress ratio, $k = \sigma_0/p_0$ = 1.1

Yield stress ratio, $k_t = p_t/p_0$ = 0.1

Hardening curve (from uniaxial compression):

Yield stress	Plastic strain
2.200E5	0.0
2.465E5	0.1
2.729E5	0.2
2.990E5	0.3
3.245E5	0.4
3.493E5	0.5
3.733E5	0.6
3.962E5	0.7
4.180E5	0.8
4.387E5	0.9
4.583E5	1.0
4.938E5	1.2
5.248E5	1.4
5.515E5	1.6
5.743E5	1.8
5.936E5	2.0
6.294E5	2.5
6.520E5	3.0
6.833E5	5.0
6.883E5	10.0

Initial conditions

Initial volumetric compacting plastic strain, $-\varepsilon_{\mathrm{vol}}^{pl}$, is set to 0.02 for the cases in which specifying an initial equivalent plastic strain is tested.

(The units are not important.)

Results and discussion

The results agree well with exact analytical or approximate solutions.

Input files

mfeoto3ahc.inp	Hydrostatic compression, C3D8 elements.
mfeoto3euc.inp	Uniaxial compression, C3D8 elements.
mfeoto3gsh.inp	Shear, C3D8 elements.
mfeoto3hut.inp	Uniaxial tension, C3D8 elements.
mfeoti3euc.inp	Uniaxial compression, CPE4 elements.
mfeoto3ltr.inp	Triaxial stress, CPE4 elements (inhomogeneous).
mfeoto3vlp.inp	Linear perturbation with **LOAD CASE* and hydrostatic compression, C3D8 elements.

Clay plasticity with linear elasticity

Elements tested

C3D8 C3D8R CAX4R CAX8R CPE4R

Problem description

Material 1:

Elasticity

The Young's modulus used in each test is given in the input file description. The modulus of each test is based on the average elastic stiffness of the equivalent test with porous elasticity at increments 10 and 20. A direct comparison with the results documented in *Drucker-Prager plasticity with linear elasticity* is, therefore, possible.

Poisson's ratio, $\nu = 0.3$

Plasticity

Critical state slope, $M = 1.0$

Initial volumetric plastic strain, $\varepsilon_{\text{vol}}^{pl}\big|_0 = 0.4$

Cap parameter, $\beta = 0.5$ (when included; otherwise, 1.0)

Third invariant ratio, $K = 0.78$ (when included; otherwise, 1.0)

The exponential hardening curve used in *Drucker-Prager plasticity with linear elasticity* is entered in tabulated form with an initial volumetric plastic strain that corresponds to a yield surface size of either $\alpha_0 = 58.3$ or $\alpha_0 = 130.9$.

(The units are not important.)

Material 2:

Elasticity

Young's modulus, E = 18820

Poisson's ratio, ν = 0.3

Plasticity

Critical state slope, M = 1.0

Initial volumetric plastic strain, $\varepsilon_{\text{vol}}^{pl}\big|_0$ = 0.0

Cap parameter, β = 1.0

Third invariant ratio, K = 1.0

Tabulated curves are used for defining the compressive and tensile hardening.

Softening regularization

$l_c^{(m)}$ = 0.8

n_r = 2.0

f_{max} = 2.5

(The units are not important.)

Material 3:

Elasticity

Young's modulus, E = 18820

Poisson's ratio, ν = 0.3

Plasticity

Critical state slope, M = 1.0

Initial volumetric plastic strain, $\varepsilon_{\text{vol}}^{pl}\big|_0$ = 0.0

Cap parameter, β = 1.0

Third invariant ratio, K = 1.0

Tabulated curves are used for defining the compressive and tensile hardening.

Softening regularization

$l_c^{(m)}$ = 0.5

n_r = 1.0

f_{max} = 2.5

(The units are not important.)

Material 4 [Crook et al. (2002)]:

Elasticity

Engineering constants	
E_1	200000.0
E_2	342000.0
E_3	342000.0
ν_{12}	0.32
ν_{13}	0.32
ν_{23}	0.32
G_{12}	89900.0
G_{13}	89900.0
G_{23}	129545.5

Plasticity

Critical state slope, M = 1.0

Initial volumetric plastic strain, $\varepsilon_{\text{vol}}^{pl}\big|_0$ = 0.0

Cap parameter, β = 1.0

Third invariant ratio, K = 1.0

Tabulated curves are used for defining the compressive and tensile hardening.

Anisotropic yield ratios: 1.2, 1.0, 1.0, 0.71, 0.71, 0.99

(The units are not important.)

Results and discussion

The results agree well with exact analytical or approximate solutions.

Input files

Abaqus/Standard input files

Material 1:

mceooo3ahc.inp	Hydrostatic compression, C3D8 elements, E = 18820.
mceoot3ctc.inp	Triaxial compression, CAX8R elements, E = 30732.
mceobo3ctc.inp	β = 0.5, triaxial compression, CAX8R elements, E = 29556.

mceooo3dte.inp Triaxial extension, CAX8R elements, E = 21114.

mcekod3dte.inp K = 0.78, triaxial extension, CAX8R elements, E = 28140.

mcekbo3dte.inp β = 0.5, K = 0.78, triaxial extension, CAX8R elements, E = 27580.

mceooo3euc.inp Uniaxial compression, CAX8R elements, E = 30000.

mceooo3gsh.inp Shear, C3D8 elements, E = 2798.

mceooo3vlp.inp Linear perturbation with **LOAD CASE* and hydrostatic compression, C3D8 elements, E = 18820.

Material 2:

clayteninipssoft3d.inp Hydrostatic compression, C3D8 and C3D8R elements.

clayteninipssoftgpe.inp Hydrostatic compression, CAX4R elements.

Material 4:

clayorthelaspoten3d.inp Uniaxial compression and shear, C3D8R elements.

clayorthelaspotengpe.inp Shear, CPE4R elements.

Abaqus/Explicit input files

Material 1:

mceooo3ahc_xpl.inp Hydrostatic compression, C3D8 elements, E = 18820.

mceoot3ctc_xpl.inp Triaxial compression, CAX4R elements, E = 30732.

mceobo3ctc_xpl.inp β = 0.5, triaxial compression, CAX4R elements, E = 29556.

mceooo3dte_xpl.inp Triaxial extension, CAX4R elements, E = 21114.

mcekod3dte_xpl.inp K = 0.78, triaxial extension, CAX4R elements, E = 28140.

mcekbo3dte_xpl.inp β = 0.5, K = 0.78, triaxial extension, CAX4R elements, E = 27580.

mceooo3euc_xpl.inp Uniaxial compression, CAX4R elements, E = 30000.

mceooo3gsh_xpl.inp Shear, C3D8 elements, E = 2798.

Material 2:

clayteninipssoft3d_xpl.inp Hydrostatic compression, C3D8 and C3D8R elements.

Material 3:

clayteninipssoftgpe_xpl.inp Hydrostatic compression, CAX4R elements.

Material 4:

clayorthelaspoten3d_xpl.inp Uniaxial compression and shear, C3D8R elements.

clayorthelaspotengpe_xpl.inp Shear, CPE4R elements.

Transferring results from Abaqus/Standard to Abaqus/Explicit

Material 1:

mceooo3ahc_sx_s.inp Abaqus/Standard analysis, hydrostatic compression, C3D8 elements, E = 18820.

mceooo3ahc_sx_x.inp Abaqus/Explicit import analysis, UPDATE=NO, STATE=YES.

mceoot3ctc_sx_s.inp Abaqus/Standard analysis, triaxial compression, CAX4R elements, E =27580.

mceoot3ctc_sx_x.inp Abaqus/Explicit import analysis, UPDATE=NO, STATE=YES.

mceobo3ctc_sx_s.inp	Abaqus/Standard analysis, $\beta = 0.5$, triaxial compression, CAX4R elements, E =29556.
mceobo3ctc_sx_x.inp	Abaqus/Explicit import analysis, UPDATE=NO, STATE=YES.
mceooo3dte_sx_s.inp	Abaqus/Standard analysis, triaxial extension, CAX4R elements, E =21114.
mceooo3dte_sx_x.inp	Abaqus/Explicit import analysis, UPDATE=NO, STATE=YES.
mcekod3dte_sx_s.inp	Abaqus/Standard analysis, $K = 0.78$, triaxial extension, CAX4R elements, E =28140.
mcekod3dte_sx_x.inp	Abaqus/Explicit import analysis, UPDATE=NO, STATE=YES.
mcekbo3dte_sx_s.inp	Abaqus/Standard analysis, $\beta = 0.5$, $K = 0.78$, triaxial extension, CAX4R elements, E =27580.
mcekbo3dte_sx_x.inp	Abaqus/Explicit import analysis, UPDATE=NO, STATE=YES.
mceooo3euc_sx_s.inp	Abaqus/Standard analysis, uniaxial compression, CAX4R elements, E =30000.
mceooo3euc_sx_x.inp	Abaqus/Explicit import analysis, UPDATE=NO, STATE=YES.
mceooo3gsh_sx_s.inp	Abaqus/Standard analysis, shear, C3D8 elements, E =2798.
mceooo3gsh_sx_x.inp	Abaqus/Explicit import analysis, UPDATE=NO, STATE=YES.

Transferring results from Abaqus/Explicit to Abaqus/Standard

Material 1:

mceoot3ctc_xs_x.inp	Abaqus/Explicit analysis, triaxial compression, CAX4R elements, E =27580.
mceoot3ctc_xs_s.inp	Abaqus/Standard import analysis, UPDATE=NO, STATE=YES.
mceooo3dte_xs_x.inp	Abaqus/Explicit analysis, triaxial extension, CAX4R elements, E =21114.
mceooo3dte_xs_s.inp	Abaqus/Standard import analysis, UPDATE=NO, STATE=YES.
mceooo3gsh_xs_x.inp	Abaqus/Explicit analysis, shear, C3D8 elements, E =2798.
mceooo3gsh_xs_s.inp	Abaqus/Standard import analysis, UPDATE=NO, STATE=YES.
mceobo3ctc_xs_x.inp	Abaqus/Explicit analysis, $\beta = 0.5$, triaxial compression, CAX4R elements, E =29556.
mceobo3ctc_xs_s.inp	Abaqus/Standard import analysis, UPDATE=NO, STATE=YES.
mcekod3dte_xs_x.inp	Abaqus/Explicit analysis, $K = 0.78$, triaxial extension, CAX4R elements, E =28140.
mcekod3dte_xs_s.inp	Abaqus/Standard import analysis, UPDATE=NO, STATE=YES.
mceooo3ahc_xs_x.inp	Abaqus/Explicit analysis, hydrostatic compression, C3D8 elements, E = 18820.
mceooo3ahc_xs_s.inp	Abaqus/Standard import analysis, UPDATE=NO, STATE=YES.

References

- Crook, A. J. L., J. G. Yu, and S. M. Willson, “Development of an Orthotropic 3D Elastoplastic Material Model for Shale,” SPE/ISRM Paper SPE 78238, 2002.

Porous metal plasticity

Elements tested

C3D8 CAX4 CAX4T CPE4

Problem description

Material:

Elasticity

Young's modulus, E = 300.0

Poisson's ratio, ν = 0.3

Plasticity

Hardening curve:

Yield stress	Plastic strain
1.0000000	0.00
1.7411011	0.05
2.7276924	0.50
2.9950454	0.80

Porous metal plasticity

Modified Gurson's model: q_1 = 1.5, q_2 = 1.0, q_3 = 2.25

(otherwise, $q_1 = q_2 = q_3$ = 1.0)

Void nucleation parameters (when included): ϵ_N = 0.3, s_N = 0.1, f_N = 0.04

Initial relative density, r_0 = 0.95 (f_0 = 0.05).

Material properties used in coupled temperature-displacement analysis:

Elasticity

Young's modulus, E = 200.0E9

Poisson's ratio, ν = 0.3

Plasticity

Hardening curve:

Yield stress	Plastic strain
7.0E8	0.00
3.7E9	10.0

Porous metal plasticity

$q_1 = q_2 = q_3 = 1.0$

Initial relative density, $r_0 = 0.95$ ($f_0 = 0.05$).

Thermal properties

Specific heat, $c_p = 586.0$

Density, $\rho = 7833.0$

Conductivity, $k = 52.0$

Coefficient of expansion, $\alpha = 1.2E{-}5$

General: (The units are not important.)

Results and discussion

The results agree well with exact analytical or approximate solutions.

Input files

mgrono2xmx.inp	Inhomogeneous deformation, displacement control, CPE4 elements.
mgrono2xmx1.inp	Same as mgrono2xmx.inp except that the initial relative density is specified using the **INITIAL CONDITIONS*, TYPE = RELATIVE DENSITY option.
mgrono3hut.inp	Uniaxial tension, traction control, nucleation of voids, C3D8 elements.
mgrono3jht.inp	Hydrostatic tension, displacement control, nucleation of voids, C3D8 elements.
mgrooo2bus.inp	Uniaxial strain (confined compression), traction control, CAX4 elements.
mgrooo2euc.inp	Uniaxial compression, traction control, CAX4 elements.
mgrooo2gsh.inp	Shear, CPE4 elements.
mgrooo2hut.inp	Uniaxial tension, displacement control, CAX4 elements.
mgrooo2jht.inp	Hydrostatic tension, displacement control, CAX4 elements.
mgrooo3gsh.inp	Shear, C3D8 elements.
mgrooo3jht.inp	Hydrostatic tension, displacement control, C3D8 elements.
mgrqno2hut.inp	Modified Gurson's model, uniaxial tension, displacement control, nucleation of voids, CAX4 elements.
mgrqoo2ahc.inp	Modified Gurson's model, hydrostatic compression, displacement control, CAX4 elements.
mgtooo2hut.inp	Uniaxial tension, coupled temperature-displacement, CAX4T elements.
mgroob2hut.inp	Uniaxial tension, displacement control, CAX4 elements, temperature and field variable dependencies.
mgrqnt2hut.inp	Modified Gurson's model, uniaxial tension, nucleation of voids, temperature dependencies.

Mohr-Coulomb plasticity

Elements tested

C3D8 C3D8R CAX4 CAX4R CPE4 CPE4R

Problem description

Material 1:

Elasticity

Young's modulus, E = 300.E3

Poisson's ratio, ν = 0.3

Plasticity

Angle of friction, ϕ = 40°

Dilation angle, ψ = 40°

Cohesion hardening curve:

Yield stress	Plastic strain
6.0E3	0.000000
9.0E3	0.020000
11.0E3	0.063333
12.0E3	0.110000
12.0E3	1.000000

Tension cutoff

Perfectly plastic, yield stress = 600.0

(The units are not important.)

Material 2:

Elasticity

Young's modulus, E = 300.E3

Poisson's ratio, ν = 0.3

Plasticity

Angle of friction, ϕ = 30°

Dilation angle, ψ = 20°

Cohesion hardening curve:

Yield stress	Plastic strain
866.025	0.0
1732.05	1.0

Tension cutoff

Softening response:

Yield stress	Plastic strain
1000.0	0.0
100.0	1.0

(The units are not important.)

Material 3:

Elasticity

Young's modulus, $E = 2$ .E7

Poisson's ratio, $\nu = 0.3$

Plasticity

Angle of friction, $\phi = 30°$

Dilation angle, $\psi = 20°$

Perfectly plastic cohesion:

Yield stress	Plastic strain
1000.0	0.0
1000.0	1.0

Tension cutoff

Perfectly plastic:

Yield stress	Plastic strain
1000.0	0.0
1000.0	1.0

(The units are not important.)

Results and discussion

The results agree well with exact analytical or approximate solutions.

Input files

Abaqus/Standard input files

Material 1:

mmoooo3jht.inp	Hydrostatic tension, C3D8 elements.
mmoooo3ltr.inp	Triaxial stress, CPE4 elements (inhomogeneous).
mmoooo3dte.inp	Triaxial extension, CAX4 elements.
mmoooo3hut.inp	Uniaxial tension, C3D8 elements.
mmoooo3gsh.inp	Shear, C3D8 elements.
mmoooo3ctc.inp	Triaxial compression, CAX4 elements.
mmoooo3euc.inp	Uniaxial compression, C3D8 elements.
mmooot3euc.inp	Uniaxial compression with temperature dependence, C3D8 elements.
mmoooo3vlp.inp	Linear perturbation steps containing **LOAD CASE*, uniaxial compression, C3D8 elements.
mctc_trxs.inp	Triaxial extension with tension cutoff, CAX4 elements.

Material 2:

mctc_ucut.inp	Tension cutoff, uniaxial compression followed by uniaxial tension, C3D8R and CAX4R elements.
mctc_psss.inp	Tension cutoff, plane strain compression/tension and simple shear, CPE4R elements.

Material 3:

mctc_btbc.inp	Tension cutoff, biaxial tension followed by biaxial compression, C3D8R element.
mctc_ptpc.inp	Tension cutoff, hydrostatic tension followed by hydrostatic compression, C3D8R element.

Abaqus/Explicit input files

Material 1:

mmoooo3jht_xpl.inp	Hydrostatic tension, C3D8 elements.
mmoooo3ltr_xpl.inp	Triaxial stress, CPE4R elements (inhomogeneous).
mmoooo3dte_xpl.inp	Triaxial extension, CAX4R elements.
mmoooo3hut_xpl.inp	Uniaxial tension, C3D8 elements.
mmoooo3gsh_xpl.inp	Shear, C3D8 elements.
mmoooo3ctc_xpl.inp	Triaxial compression, CAX4R elements.
mmoooo3euc_xpl.inp	Uniaxial compression, C3D8 elements.
mmooot3euc_xpl.inp	Uniaxial compression with temperature dependence, C3D8 elements.
mctc_trxs_xpl.inp	Triaxial extension with tension cutoff, CAX4R elements.

Material 2:

mctc_ucut_xpl.inp	Tension cutoff, uniaxial compression followed by uniaxial tension, C3D8R and CAX4R elements.
mctc_psss_xpl.inp	Tension cutoff, plane strain compression/tension and simple shear, CPE4R elements.

Transferring results from Abaqus/Standard to Abaqus/Explicit

Material 1:

mmoooo3jht_sx_s.inp	Abaqus/Standard analysis, hydrostatic tension, C3D8 elements.
mmoooo3jht_sx_x.inp	Abaqus/Explicit import analysis, UPDATE=NO, STATE=YES.
mmoooo3ltr_sx_s.inp	Abaqus/Standard analysis, triaxial stress, CPE4R elements.
mmoooo3ltr_sx_x.inp	Abaqus/Explicit import analysis, UPDATE=NO, STATE=YES.
mmoooo3dte_sx_s.inp	Abaqus/Standard analysis, triaxial extension, CAX4R elements.
mmoooo3dte_sx_x.inp	Abaqus/Explicit import analysis, UPDATE=NO, STATE=YES.
mmoooo3hut_sx_s.inp	Abaqus/Standard analysis, uniaxial tension, C3D8 elements.
mmoooo3hut_sx_x.inp	Abaqus/Explicit import analysis, UPDATE=NO, STATE=YES.
mmoooo3gsh_sx_s.inp	Abaqus/Standard analysis, shear, C3D8 elements.
mmoooo3gsh_sx_x.inp	Abaqus/Explicit import analysis, UPDATE=NO, STATE=YES.
mmoooo3ctc_sx_s.inp	Abaqus/Standard analysis, triaxial compression, CAX4R elements.
mmoooo3ctc_sx_x.inp	Abaqus/Explicit import analysis, UPDATE=NO, STATE=YES.
mmoooo3euc_sx_s.inp	Abaqus/Standard analysis, uniaxial compression, C3D8 elements.
mmoooo3euc_sx_x.inp	Abaqus/Explicit import analysis, UPDATE=NO, STATE=YES.

Material 3:

sx_s_mctc.inp	Abaqus/Standard analysis, uniaxial tension followed by compression, C3D8R element.
sx_x_mctc_n_y.inp	Abaqus/Explicit import analysis, UPDATE=NO, STATE=YES.

Transferring results from Abaqus/Explicit to Abaqus/Standard

Material 1:

mmoooo3dte_xs_x.inp	Abaqus/Explicit analysis, triaxial extension, CAX4R elements.
mmoooo3dte_xs_s.inp	Abaqus/Standard import analysis, UPDATE=NO, STATE=YES.
mmoooo3gsh_xs_x.inp	Abaqus/Explicit analysis, shear, C3D8 elements.
mmoooo3gsh_xs_s.inp	Abaqus/Standard import analysis, UPDATE=NO, STATE=YES.
mmoooo3ctc_xs_x.inp	Abaqus/Explicit analysis, triaxial compression, CAX4R elements.
mmoooo3ctc_xs_s.inp	Abaqus/Standard import analysis, UPDATE=NO, STATE=YES.
mmoooo3ltr_xs_x.inp	Abaqus/Explicit analysis, triaxial stress, CPE4R elements.
mmoooo3ltr_xs_s.inp	Abaqus/Standard import analysis, UPDATE=NO, STATE=YES.
mmoooo3jht_xs_x.inp	Abaqus/Explicit analysis, hydrostatic tension, C3D8 elements.
mmoooo3jht_xs_s.inp	Abaqus/Standard import analysis, UPDATE=NO, STATE=YES.
mmoooo3hut_xs_x.inp	Abaqus/Explicit analysis, uniaxial tension, C3D8 elements.
mmoooo3hut_xs_s.inp	Abaqus/Standard import analysis, UPDATE=NO, STATE=YES.

Material 3:

xs_x_mctc.inp	Abaqus/Explicit analysis, uniaxial tension, C3D8R element.
xs_s_mctc_n_y.inp	Abaqus/Standard import analysis, UPDATE=NO, STATE=YES.

Cast iron plasticity

Elements tested

C3D8 CAX4 CAX4T CPE4 T3D2

Problem description

Material:

Elasticity

Young's modulus, E = 14.773E6

Poisson's ratio, ν = 0.2273

Plasticity

Plastic “Poisson's ratio,” ν_{pl} = 0.039

Hardening curves: The hardening curves in tension and compression are illustrated in *Figure 1*.

Thermal properties

Specific heat, c_p = 47.52

Density, ρ = 439.92

Conductivity, k = 9.4

Coefficient of expansion, α = 11.0E−6

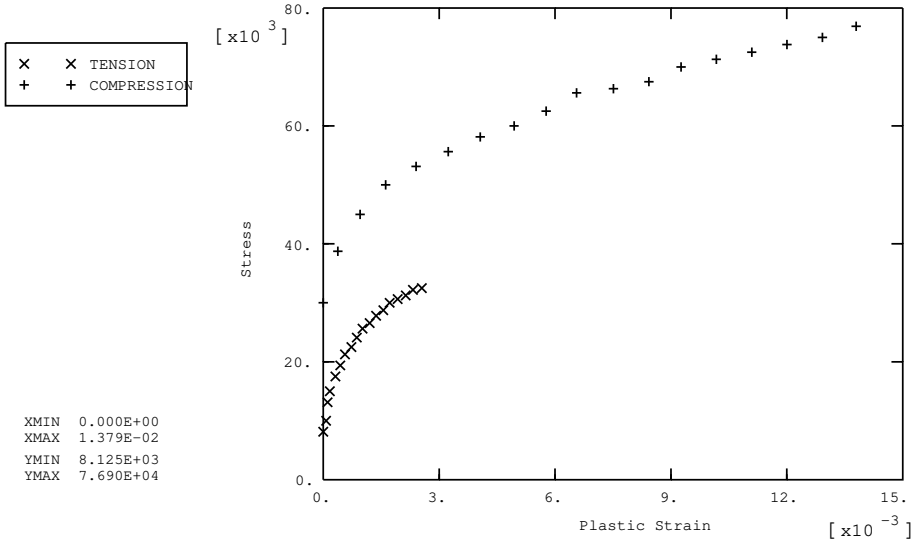


Figure 1: Stress versus plastic strain under uniaxial tension and uniaxial compression.

(The units are not important.)

Results and discussion

Most tests in this section are set up as cases of the homogeneous deformation of a single element of unit dimensions. Consequently, the results are identical for all integration points within the element.

Input files

Abaqus/Standard input files

mciooo3jht.inp	Hydrostatic tension, C3D8 elements.
mciooo3gsh.inp	Shear, C3D8 elements.
mciooo3hut.inp	Uniaxial tension, CAX4 elements.
mcioot3euc.inp	Uniaxial compression with temperature dependence, C3D8 elements.
mctoot3hut.inp	Uniaxial tension, coupled temperature-displacement, CAX4T elements.
mciooo3xmx.inp	Inhomogeneous deformation, CPE4 elements.
mciooo1hut.inp	Uniaxial tension and linear perturbation steps containing **LOAD CASE*, T3D2 elements.

Abaqus/Explicit input files

mciooo3jht_xpl.inp	Hydrostatic tension, C3D8 elements.
mciooo3gsh_xpl.inp	Shear, C3D8 elements.
mciooo3hut_xpl.inp	Uniaxial tension, CAX4R elements.
mcioot3euc_xpl.inp	Uniaxial compression with temperature dependence, C3D8 elements.
mciooo3xmx_xpl.inp	Inhomogeneous deformation, CPE4 elements.
mciooo1hut_xpl.inp	Uniaxial tension, T3D2 elements.

Transferring results from Abaqus/Standard to Abaqus/Explicit

mciooo3gsh_sx_s.inp	Abaqus/Standard analysis, shear, C3D8 elements.
mciooo3gsh_sx_x.inp	Abaqus/Explicit import analysis from mciooo3gsh_sx_s.inp.
mciooo3gsh_xs_s.inp	Abaqus/Standard import analysis from mciooo3gsh_sx_x.inp.
mciooo3gsh_xx_x2.inp	Abaqus/Explicit import analysis from mciooo3gsh_sx_x.inp.

Soft rock plasticity with porous elasticity

Elements tested

C3D8R , CPE4R

Problem description

Material 1:

Power law–based porous elasticity

E_{ref} = 75000.0

p_{ref} = 1.0

p_0 = 1.0

n = 0.

$\nu_\infty = 0.2$

$\nu_0 = 0.2$

$m = 1.0$

Plasticity

$\beta = 60.0$

$\psi = 50.0$

$n_y = 1.6$

$f_0 = 0.001$

$f_1 = 0.0007$

$\alpha = 0.25$

$\varepsilon_{\text{vol}}^{pl}\big|_0 = 0.05$

Tabulated curves are used for defining the compressive and tensile hardening. (The units are not important.)

Material 2:

Power law–based porous elasticity

$E_{ref} = 75000.0$

$p_{ref} = 1.0$

$p_0 = 1.0$

$n = 0.$

$\nu_\infty = 0.2$

$\nu_0 = 0.2$

$m = 1.0$

Plasticity

$\beta = 45.0$

$\psi = 45.0$

$n_y = 1.$

$f_0 = 0.001$

$f_1 = 0.0007$

$\alpha = 0.0$

$\varepsilon_{\text{vol}}^{pl}\big|_0 = 0.0$

Tabulated curves are used for defining the compressive and tensile hardening.
(The units are not important.)

Results and discussion

The results agree well with exact analytical or approximate solutions.

Input files

softrockporelatests.inp	Hydrostatic compression, uniaxial compression, and shear using materials 1 and 2.

Soft rock plasticity with linear elasticity

Elements tested
C3D8R , CPE4R

Problem description

Material 1:

Linear elasticity

Young's modulus, $E = 75.E3$

Poisson's ratio, $\nu = 0.2$

Plasticity

$\beta = 60.0$

$\psi = 50.0$

$n_y = 1.6$

$f_0 = 0.001$

$f_1 = 0.0007$

$\alpha = 0.25$

$\varepsilon_{\text{vol}}^{pl}\big|_0 = 0.05$

Tabulated curves are used for defining the compressive and tensile hardening.
(The units are not important.)

Material 2:

Linear elasticity

Young's modulus, E = 75.E3

Poisson's ratio, ν = 0.2

Plasticity

β = 45.0

ψ = 45.0

n_y = 1.

f_0 = 0.001

f_1 = 0.0007

α = 0.0

$\varepsilon^{pl}_{vol}\big|_0$ = 0.0

Tabulated curves are used for defining the compressive and tensile hardening.
(The units are not important.)

Results and discussion

The results agree well with exact analytical or approximate solutions.

Input files

Abaqus/Standard input files

softrocklineelatests.inp	Hydrostatic compression, uniaxial compression, and shear using materials 1 and 2.

Abaqus/Explicit input files

softrocklineelatests_xpl.inp	Hydrostatic compression, uniaxial compression, and shear using materials 1 and 2.

Plasticity model for superelastic materials

Elements tested
C3D8 , CPS4 , CPS4R

Problem description

Material:

Elasticity

E_A = 40000.0

ν_A = 0.33

E_M = 32000.0

ν_M = 0.33

ε^L = 0.041

σ^S_{tL} = 440.0

σ^E_{tL} = 540.0

σ^S_{tU} = 250.0

σ^E_{tU} = 140.0

σ^S_{cL} = 440.0

T_0 = 22.0

$\left(\frac{\delta\sigma}{\delta T}\right)_L$ = 6.7

$\left(\frac{\delta\sigma}{\delta T}\right)_U$ = 6.7

Plasticity

Tabulated curves are used for defining the yield stress as a function of total strain. (The units are not important.)

Results and discussion

The results agree well with exact analytical or approximate solutions.

Input files

Abaqus/Standard input files

superelas_std.inp Tension test.

Abaqus/Explicit input files

superelas_xpl.inp Tension test.

Rate-dependent plasticity in Abaqus/Standard

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Rate-dependent Mises plasticity

Elements tested

C3D8

Problem description

Material:

Elasticity

Young's modulus, E=200.0E3

Poisson's ratio, ν=0.3

Plasticity

Hardening:

Yield stress	Plastic strain
200.	0.0000
220.	0.0009
220.	0.0029

Rate dependence parameter, D=40.0

Rate dependence parameter, p=5.0

The rate dependence parameters are as follows for the test that verifies the temperature dependencies:

D=30.0, p=3.0 at 10.0°

D=50.0, p=7.0 at 20.0°

The power law is entered as a piecewise linear relationship for the cases in which rate-dependent test data are specified directly.

(The units are not important.)

Results and discussion

The results agree well with exact analytical or approximate solutions.

Input files

mprooo3hut.inp	Uniaxial tension, power law, C3D8 elements.

mpryso3hut.inp	Uniaxial tension, yield ratios, C3D8 elements.
mproot3hut.inp	Uniaxial tension, temperature-dependent power law, C3D8 elements.
mpryst3hut.inp	Uniaxial tension, temperature-dependent yield ratios, C3D8 elements.
mprooo3vlp.inp	Linear perturbation uniaxial tension, power law, C3D8 elements.
mprpro3vlp.inp	Linear perturbation uniaxial tension, **PLASTIC*, RATE=option, C3D8 elements.

Adiabatic rate-dependent Mises plasticity

Elements tested

C3D8 T3D2

Problem description

Material:

Elasticity

Young's modulus, E=30.0E6

Poisson's ratio, ν=0.3

Plasticity

Hardening:

Yield stress	Plastic strain	Temperature
30.0E3	0.000	0.0
50.0E3	0.200	0.0
50.0E3	2.000	0.0

Other properties

Density, μ=1000.0

Specific heat, c=0.4

Inelastic heat fraction = 0.5

Rate dependence parameter, D=40.0

Rate dependence parameter, p=5.0

The power law is entered as a piecewise linear relationship for the cases in which rate-dependent test data are specified directly.

(The units are not important.)

Results and discussion

The results agree well with exact analytical or approximate solutions.

Input files

mhriho3hut.inp	Uniaxial tension, power law, C3D8 elements.
mhrpro3hut.inp	Uniaxial tension, **PLASTIC*, RATE=option, C3D8 elements.
mhriho1hut.inp	Uniaxial tension, power law, T3D2 elements.
mhryso1hut.inp	Uniaxial tension, yield ratios, T3D2 elements.
mhriho3xmx.inp	Multiaxial, power law, C3D8 elements.
mhrpro3xmx.inp	Multiaxial, **PLASTIC*, RATE=option, C3D8 elements.

Rate-dependent Hill plasticity

Elements tested

C3D8

Problem description

Material:

Elasticity

Young's modulus, E=200.0E3

Poisson's ratio, ν=0.3

Plasticity

Hardening:

Yield stress	Plastic strain
200.	0.0000
220.	0.0009
220.	0.0029

Anisotropic yield ratios: 1.5, 1.2, 1.0, 1.0, 1.0, 1.0

Rate dependence parameter, D=40.0

Rate dependence parameter, p=5.0

The power law is entered as a piecewise linear relationship for the cases in which rate-dependent test data are specified directly.

(The units are not important.)

Results and discussion

The results agree well with exact analytical or approximate solutions.

Input files

mpxooo3nt1.inp	Uniaxial tension in direction 1, power law, C3D8 elements.
mpxyso3nt1.inp	Uniaxial tension in direction 1, yield ratios, linear perturbation with **LOAD CASE*, C3D8 elements.

mpxooo3ot2.inp	Uniaxial tension in direction 2, power law, C3D8 elements.
mpxooo3pt3.inp	Uniaxial tension in direction 3, power law, C3D8 elements.
mpxpro3pt3.inp	Uniaxial tension in direction 3, **PLASTIC*, RATE=option, C3D8 elements.

Rate-dependent Drucker-Prager plasticity

Elements tested

C3D8 CPS4

Problem description

Material:

Elasticity

Young's modulus, E=300.0E3

Poisson's ratio, ν=0.3

Plasticity

The linear Drucker-Prager model is used in each case.

Angle of friction, β=40.0

Dilation angle, ψ=40.0

Rate dependence parameter, D=10.0

Rate dependence parameter, p=1.0

For the test that verifies the temperature dependencies, the rate dependence parameters are as follows:

D=9.0, p=0.9 at 10.0°

D=11.0, p=1.1 at 20.0°

Hardening curve:

Yield stress	Plastic strain
6.0E3	0.000000
9.0E3	0.020000
11.0E3	0.063333
12.0E3	0.110000
12.0E3	1.000000

The power law is entered as a piecewise linear relationship for the cases in which rate-dependent test data are specified directly.

(The units are not important.)

Results and discussion

The tests in this section are set up as cases of homogeneous deformation of a single element of unit dimensions. Consequently, the results are identical for all integration points within the element. The constitutive path is integrated with 20 increments of fixed size.

Input files

mdrooo3euc.inp	Uniaxial compression, power law, C3D8 elements.
mdryso3euc.inp	Uniaxial compression, yield ratios, C3D8 elements.
mdroot3euc.inp	Uniaxial compression, temperature-dependent power law, C3D8 elements.
mdryst3euc.inp	Uniaxial compression, temperature-dependent yield ratios, C3D8 elements.
mdrooo2euc.inp	Uniaxial compression, power law, CPS4 elements.
mdryro2euc.inp	Uniaxial compression, linear perturbation with **LOAD CASE*, **DRUCKER PRAGER HARDENING*, RATE=option, CPS4 elements.
mdrooo3vlp.inp	Linear perturbation uniaxial compression, power law, C3D8 elements.
mdryso3vlp.inp	Linear perturbation uniaxial compression, yield ratios, C3D8 elements.

Rate-dependent crushable foam plasticity

Elements tested

C3D8

Problem description

Material:

Elasticity

Young's modulus, E=3.0E6

Poisson's ratio, ν=0.2

Plasticity

Initial yield stress in hydrostatic compression, p_0=2.0E5

Strength in hydrostatic tension, p_t=2.0E4

Initial yield stress in uniaxial compression, σ_0=2.2E5

Yield stress ratio, $k = \sigma_0/p_0$=1.1

Yield stress ratio, $k_t = p_t/p_0$=0.1

Rate dependence parameter, D=10.0

Rate dependence parameter, p=1.0

Hardening curve (from uniaxial compression):

Yield stress	plastic strain
2.200E5	0.0

Yield stress	plastic strain
2.465E5	0.1
2.729E5	0.2
2.990E5	0.3
3.245E5	0.4
3.493E5	0.5
3.733E5	0.6
3.962E5	0.7
4.180E5	0.8
4.387E5	0.9
4.583E5	1.0
4.938E5	1.2
5.248E5	1.4
5.515E5	1.6
5.743E5	1.8
5.936E5	2.0
6.294E5	2.5
6.520E5	3.0
6.833E5	5.0
6.883E5	10.0

For the test that verifies the temperature dependencies, the rate dependence parameters are as follows:

D=9.0, p=0.9 at 10.0°

D=11.0, p=1.1 at 20.0°

The power law is entered as a piecewise linear relationship for the cases in which rate-dependent test data are specified directly.

(The units are not important.)

Results and discussion

The results agree well with exact analytical or approximate solutions.

Input files

mfrooo3euc.inp	Uniaxial compression, power law, C3D8 elements.
mfryso3euc.inp	Uniaxial compression, yield ratios, C3D8 elements.
mfroot3euc.inp	Uniaxial compression, linear perturbation with **LOAD CASE*, temperature-dependent power law, C3D8 elements.
mfryst3euc.inp	Uniaxial compression, temperature-dependent yield ratios, elements.

Rate-dependent porous metal plasticity

Elements tested

C3D8

Problem description

Material:

Elasticity

Young's modulus, E=200.0E3

Poisson's ratio, ν=0.3

Plasticity

Hardening curve:

Yield stress	Plastic strain
200.0	0.0000
220.0	0.0009
220.0	0.0029

Rate dependence parameter, D=40.0

Rate dependence parameter, p=5.0

Porous metal plasticity

q_1=q_2=q_3=1.0

Initial relative density, r_0=0.95 (f_0=0.05).

(The units are not important.)

Results and discussion

The results agree well with exact analytical or approximate solutions.

Input files

mgrooo3vlp.inp	Uniaxial tension, power law, C3D8 elements.
mgrpro3vlp.inp	Uniaxial tension; **PLASTIC*, RATE=option; C3D8 elements; linear perturbation with **LOAD CASE*.
mgryso3hut.inp	Hydrostatic tension, yield ratios, C3D8 elements.

Rate-dependent plasticity in Abaqus/Explicit

Products: Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Rate-dependent Mises plasticity

Elements tested

CPE4R CPS4R C3D8R

Features tested

Mises plasticity, rate dependence.

Problem description

This problem is a one-element verification problem for Mises plasticity with rate dependence. Three different element types are tested by stretching the element in the global *y*-direction. *Figure 1* shows the eight elements used in the analysis. The 8-node brick element (C3D8R) appears twice. The plane stress instance has no boundary conditions applied to the out-of-plane direction, and the element should respond in a state of plane stress, except for some dynamic oscillations. The plane strain instance has zero displacement boundary conditions applied to all out-of-plane displacements, and the element should respond in a state of plane strain.

The bottom and top nodes of each element are given equal and opposite prescribed velocities (v, ramping up from 0 to v_{max}) in the y-direction. The original length of each side of the elements is $L_0 = 1$. The nominal strain rate is, therefore, $2v$, with its maximum value being $2v_{max}$. The plasticity model in elements 1 through 4 in *Figure 1* has no rate dependence. The plasticity model in elements 5 through 8 is rate dependent.

This analysis is run with maximum strain rates of 2, 20, and 200 sec^{-1}.

Results and discussion

Figure 2 shows the deformed mesh at the maximum displacement. This corresponds to a nominal strain of 100%.

Figure 3 contains plots of nominal strain versus Mises stress at different strain rates for the plane strain cases. The names of the individual curves that appear in the graph legend are a concatenation of an element model type, an underscore (_), and the element numbers. The results obtained with the 8-node brick element are identical to those obtained for the 4-node quadrilateral at all strain rates. There are 12 curves plotted in the figure. For the three velocity values the two element types (CPE4R and C3D8R) are plotted using the rate-dependent and rate-independent results. The velocities vary by an order of magnitude in each case, and the number of explicit time increments used also varies by an order of magnitude. The rate-independent results are plotted for each velocity case to verify that the rate-independent plasticity integration is not overly sensitive to the strain increment size.

Figure 4 contains plots of Mises stress versus nominal strain at different strain rates for the plane stress cases. The same 12 curves are plotted as for the plane strain case.

The results presented here are the same as those obtained with Abaqus/Standard.

Input files

ratedep020.inp	Strain rate of 20.
ratedep002.inp	Strain rate of 2.
ratedep200.inp	Strain rate of 200.
ratedep_tabular.inp	Overstress power law is entered as a piecewise linear function.
ratedep_tabular_rtol.inp	Demonstrates the use of the RTOL parameter on the **MATERIAL* option.

Figures

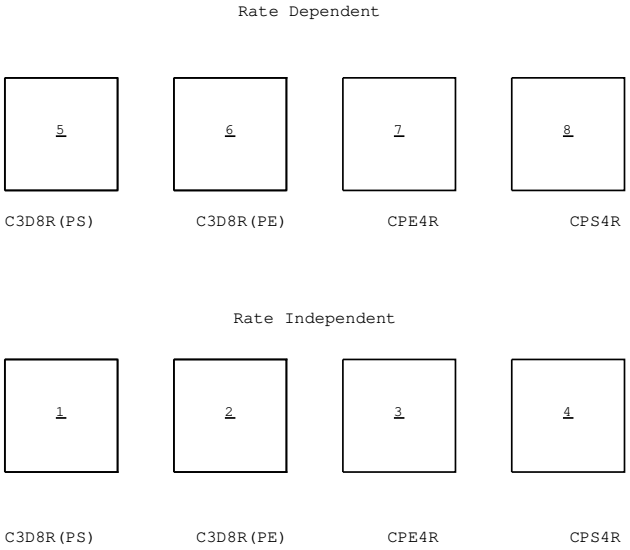


Figure 1: Original mesh for one-element rate-dependent plasticity tests.

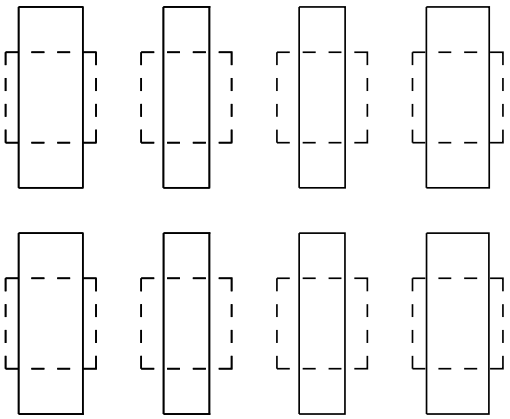


Figure 2: Total deformation for one-element rate-dependent plasticity tests.

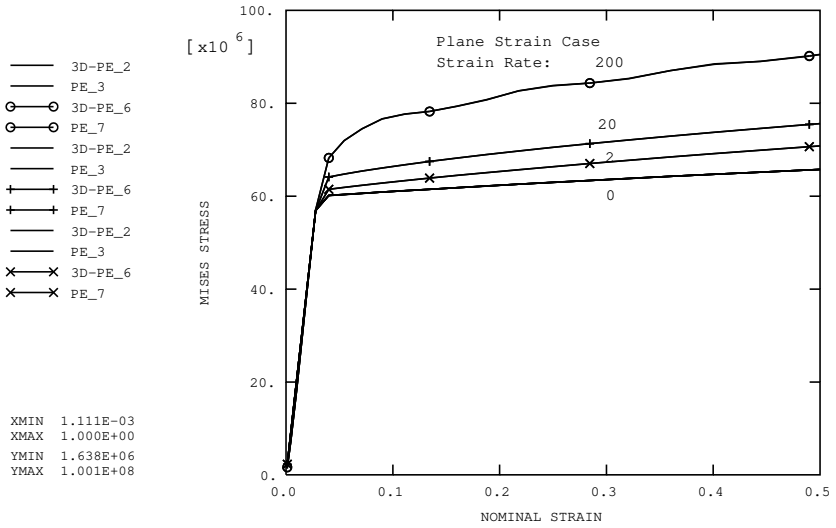


Figure 3: Mises stress versus nominal strain for plane strain cases.

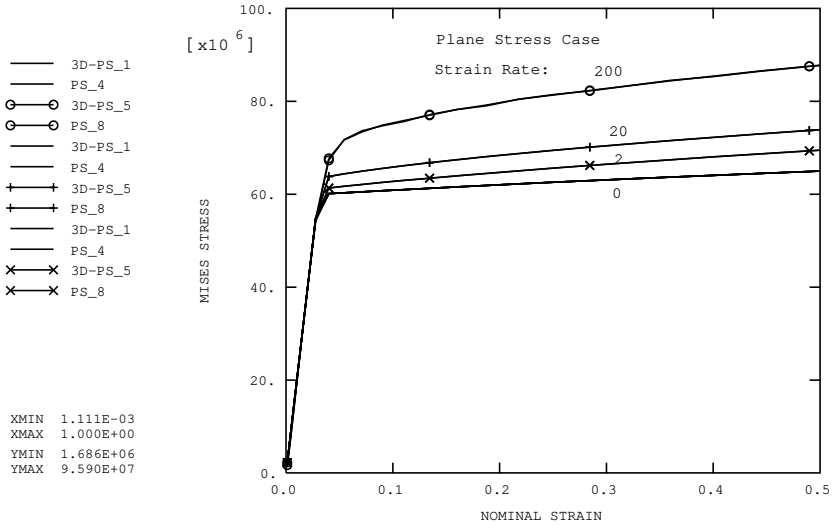


Figure 4: Mises stress versus nominal strain for plane stress cases.

Johnson-Cook rate dependence

Elements tested

C3D8R

Features tested

Johnson-Cook rate dependence in combination with Mises plasticity and Drucker-Prager plasticity.

Problem description

This problem is a one-element verification problem for Mises plasticity and Drucker-Prager plasticity in combination with Johnson-Cook strain-rate dependence. The element is subjected to uniaxial loading conditions.

Results and discussion

The results agree well with exact analytical or approximate solutions.

Input files

jcrateplasticuni.inp	Johnson-Cook rate dependence and Mises plasticity.
jcratedpexpuni.inp	Johnson-Cook rate dependence and Drucker-Prager plasticity, exponent form.
jcratedphypuni.inp	Johnson-Cook rate dependence and Drucker-Prager plasticity, hyperbolic form.
jcratedplinuni.inp	Johnson-Cook rate dependence and Drucker-Prager plasticity, linear form.

Annealing temperature

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D8 C3D8R CPS4 T3D2

Problem description

Material:

Elasticity

Young's modulus, E = 200.0E3

Poisson's ratio, ν = 0.3

Plasticity

Hardening:

Yield stress	Plastic strain	Temperature
200.	0.0000	0.0000
220.	0.0009	0.0000
220.	0.0029	0.0000
2.	0.0000	100.00

Annealing temperature

100.00

(The units are not important.)

Results and discussion

In all the tests a single element is loaded in the plastic range and then unloaded. The resulting equivalent plastic strain (and shift tensor for the kinematic models) is then annealed by raising the temperature to the annealing temperature. Subsequently, the temperature is decreased below the annealing temperature, and the element is loaded again into the plastic range.

Input files

maniht3tua.inp	Abaqus/Standard, C3D8 elements, isotropic hardening.
maniht2tua.inp	Abaqus/Standard, CPS4 elements, isotropic hardening.

mankht1tua.inp	Abaqus/Standard, T3D2 elements.
mancht3tua.inp	Abaqus/Standard, C3D8 elements, combined hardening.
mancht2tua.inp	Abaqus/Standard, CPS4 elements, combined hardening.
annealtemp.inp	Abaqus/Explicit, C3D8R elements.

Temperature-dependent inelastic materials

Products: Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

T2D2 T3D2 B21 B31 PIPE21 PIPE31 SAX1 S4R S4RS S4RSW C3D8R CPE4R CPS4R CAX4R M3D4R

Features tested

Temperature-dependent material properties with predefined field variables are tested for the following inelastic material models: Mises plasticity, Drucker plasticity, Hill's potential plasticity, crushable foam plasticity with volumetric hardening, crushable foam plasticity with isotropic hardening, ductile failure plasticity, rate-dependent Hill's potential plasticity, rate-dependent Mises plasticity, Drucker-Prager/Cap plasticity, and porous metal plasticity.

Problem description

This verification test consists of a set of single-element models that include combinations of all of the available element types with all of the available material models. All of the elements are loaded with a tensile load defined by specifying the vertical velocity at the top nodes of each element with the bottom nodes fixed. The temperature at all nodes of each element increases from an initial value of 0° to a final value of 100°. The material properties are defined as a linear function of temperature. For every material model only those element types available for the model are used. The undeformed meshes are shown in *Figure 1*, and the material properties are listed in *Table 1*.

Results and discussion

Figure 2 shows the history plot of Mises stress for the Mises plasticity model for all elements, except for pipe elements, which are consistent with beams. We can see the material softening because the yield stress drops as the temperature increases. *Figure 3* through *Figure 11* show the history plots of Mises stress for the other material models.

This problem tests the features listed but does not provide independent verification of them.

Input files

temp_plastic.inp	Input data used in this analysis.
temp_plastic_ef1.inp	External file referenced in this input.

Tables

Table 1: Material properties.

Material	Properties	T=0	T=100
Mises plasticity (density=8032)	E	193.1×10^9	160.1×10^9
	ν	0.3	0.3
	σ_0	206893	186893
	H	206893	186893
Drucker plasticity (density=1000)	E	2.1×10^7	1.9×10^7
	ν	0.3	0.3
	σ_0	40000	36000
	H	40000	39000
	β	40	39
	K	1.0	0.9
	ψ	20.0	19.0
Hill's plasticity (density=2500)	E	1.0×10^9	8.0×10^8
	ν	0.3	0.31
	σ_0	1.0×10^6	9.0×10^5
	H	4.0×10^5	3.7×10^5
Crushable foam with volumetric hardening (density=500)	E	3.0×10^6	2.0×10^6
	ν	0.0	0.0
	k	1.1	0.9
	k_t	0.1	0.1
Crushable foam with isotropic hardening (density=500)	E	3.0×10^6	2.0×10^6
	ν	0.0	0.0
	k	1.1	0.9
	ν_p	0.2983	0.10
Ductile failure (density=5800)	E	2.0×10^8	1.8×10^8
	ν	0.3	0.3
	σ_0	2.0×10^5	1.8×10^5
	H	4.0×10^5	3.8×10^5
Hill's plasticity (density=5850)	E	1.8×10^8	2.0×10^8
(rate dependent)	ν	0.3	0.3
	σ_0	1.8×10^5	1.7×10^5
	H	−8000	−8000

Material	Properties	T=0	T=100
Mises plasticity (density=1500)	E	2.0×10^9	1.8×10^9
(rate dependent)	ν	0.4	0.4
	σ_0	6.0×10^7	5.5×10^7
	H	2.0×10^7	3.5×10^7
Drucker-Prager/Cap plasticity	E	30000	29000
(density= 2.4×10^{-3})	ν	0.3	0.29
	d	100	99
	β	37.67	36.67
	R	0.1	0.11
	ϵ_{vol}^{pl}	0.0	0.0
	α	0.01	0.011
Porous metal plasticity	E	2.0×10^{11}	1.8×10^{11}
(density=7.7×10^7)	ν	0.33	0.33
	σ_0	7.5×10^8	7.5×10^8
	H	0.0	0.0

Figures

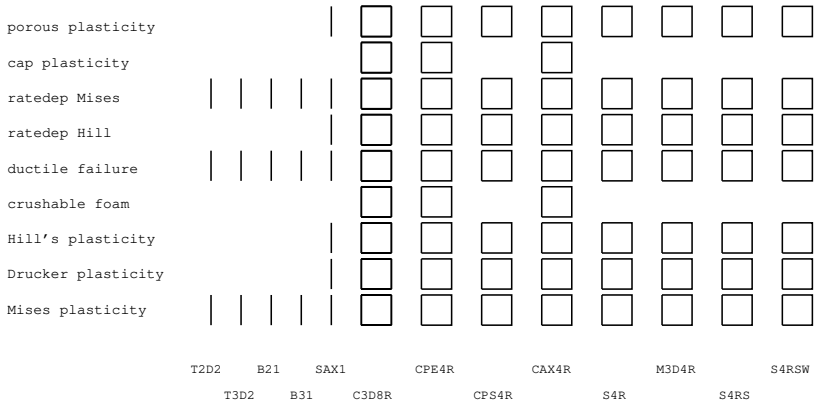


Figure 1: Temperature-dependent material property test for inelastic materials.

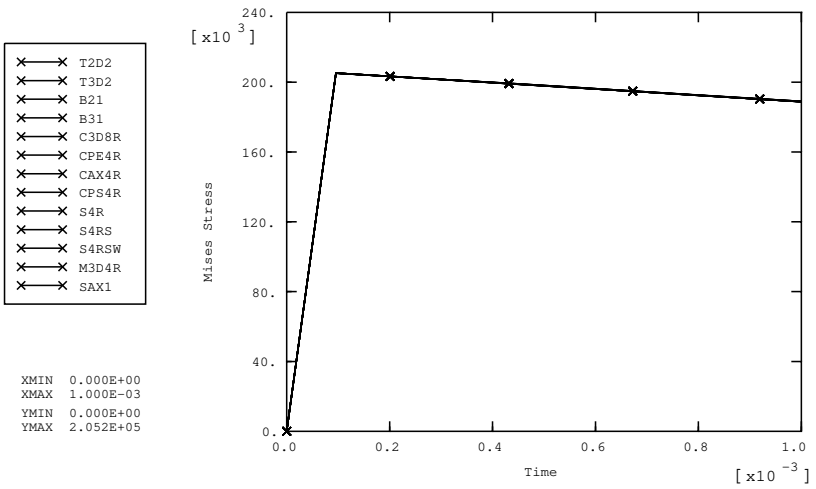


Figure 2: Mises stress versus time for Mises plasticity.

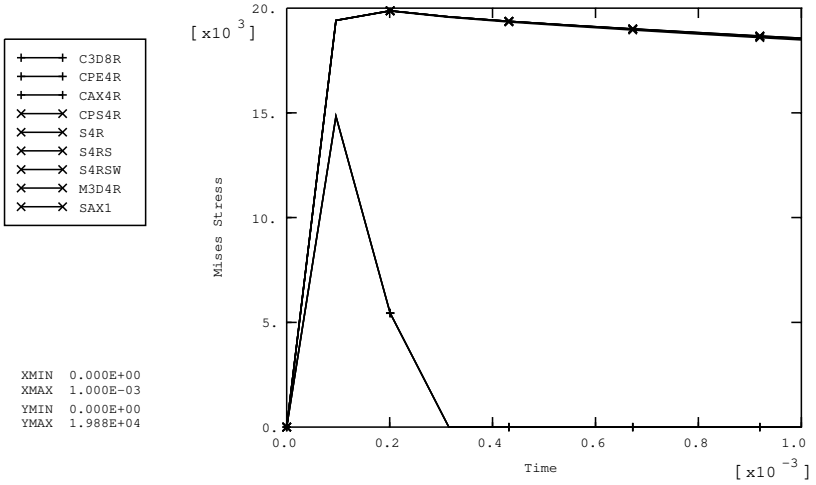


Figure 3: Mises stress versus time for Drucker plasticity.

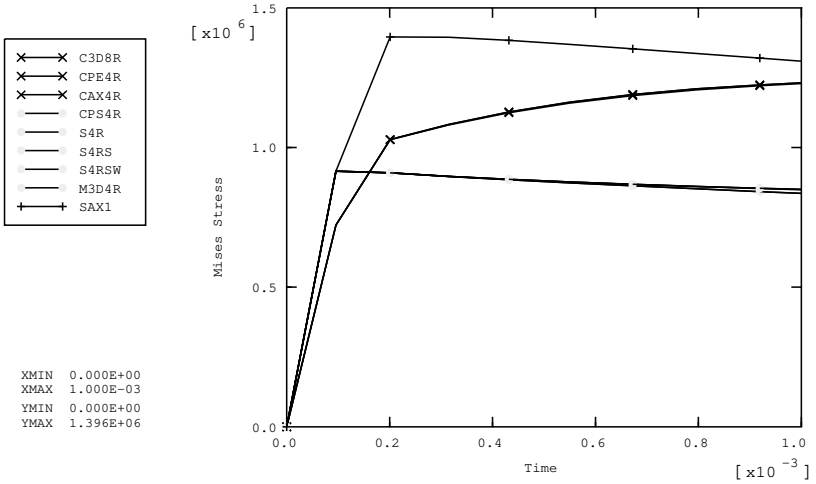


Figure 4: Mises stress versus time for Hill's plasticity.

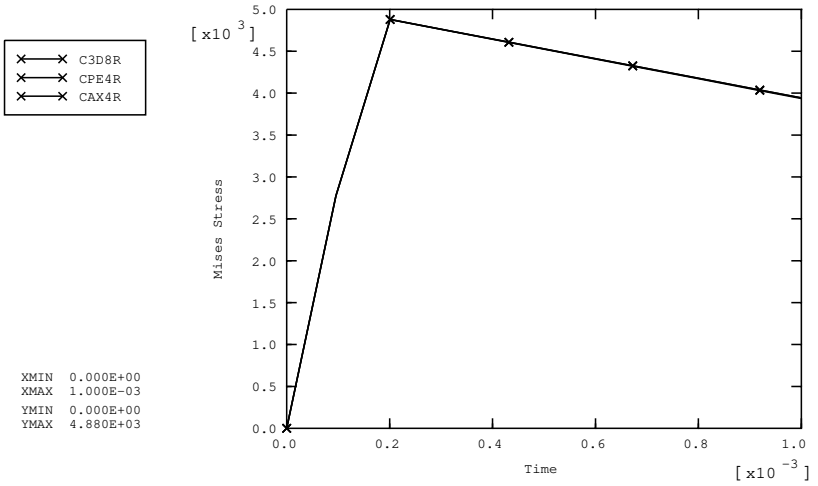


Figure 5: Mises stress versus time for crushable foam plasticity with volumetric hardening.

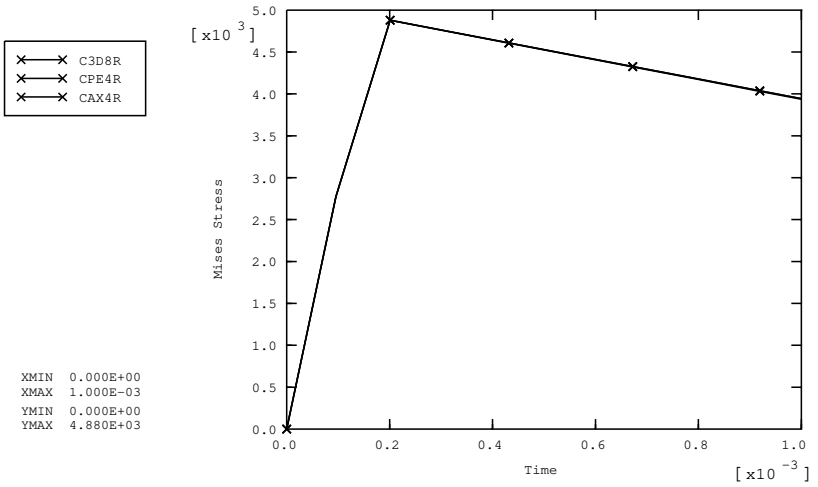


Figure 6: Mises stress versus time for crushable foam plasticity with isotropic hardening.

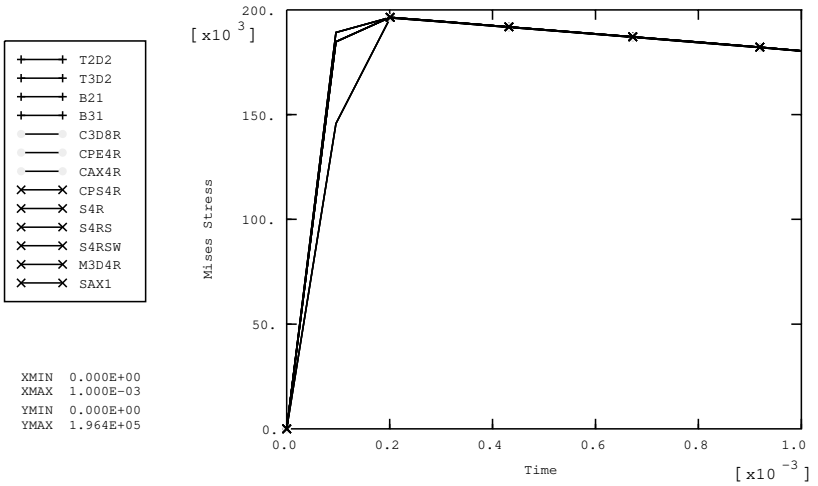


Figure 7: Mises stress versus time for ductile failure plasticity.

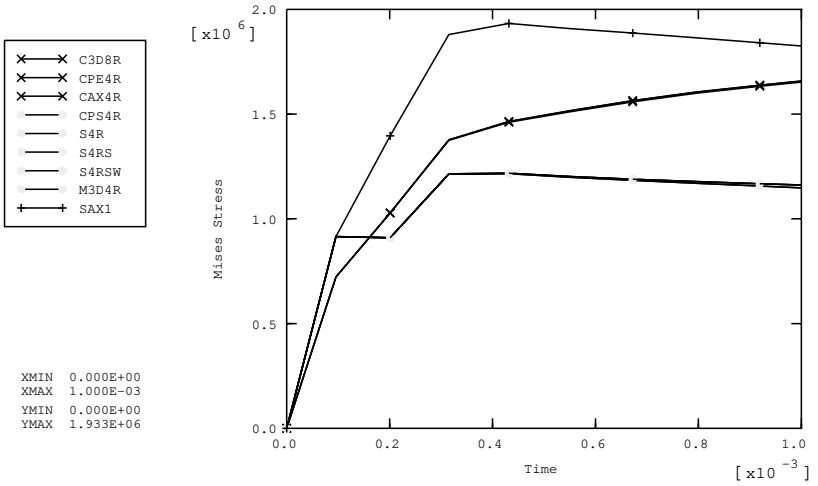


Figure 8: Mises stress versus time for rate-dependent Hill's plasticity.

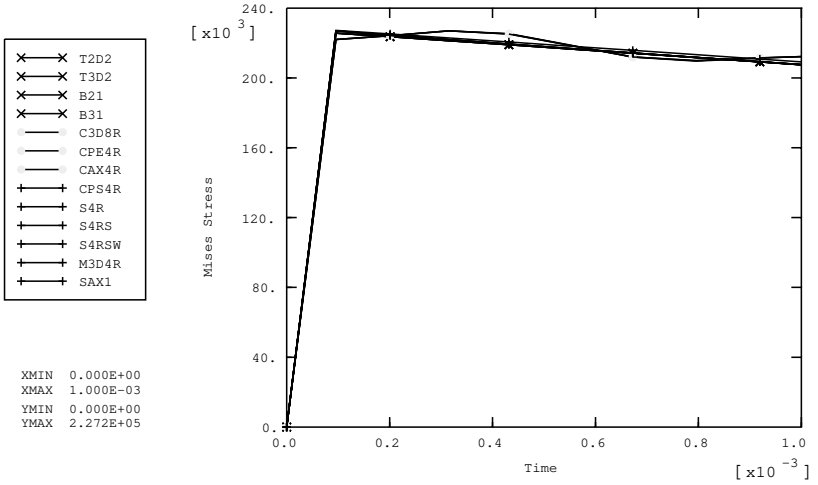


Figure 9: Mises stress versus time for rate-dependent Mises plasticity.

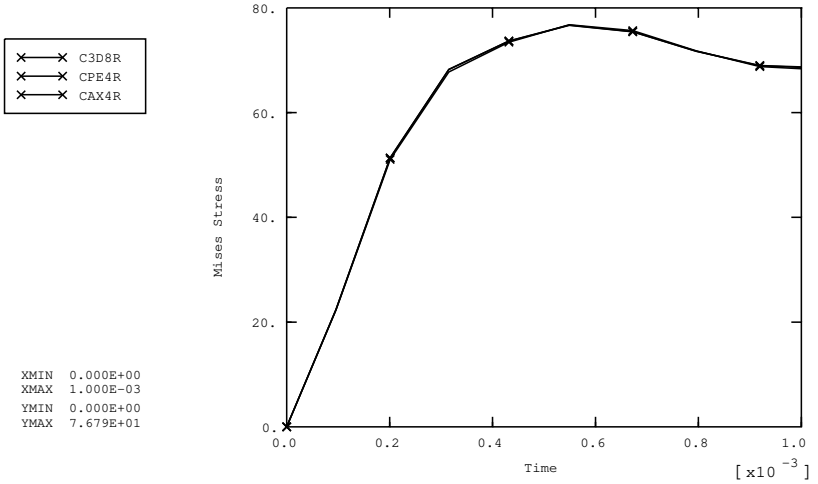


Figure 10: Mises stress versus time for Drucker-Prager/Cap plasticity.

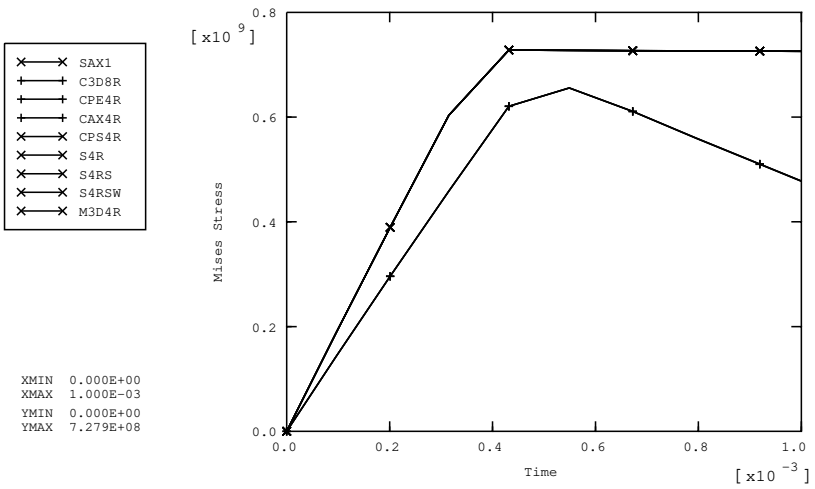


Figure 11: Mises stress versus time for porous metal plasticity.

Field-variable-dependent inelastic materials

Products: Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

T2D2 T3D2 B21 B31 PIPE21 PIPE31 SAX1 S4R S4RS S4RSW C3D8R CPE4R CPS4R CAX4R M3D4R

Features tested

Field-variable-dependent material properties with predefined temperature fields are tested for the following inelastic material models: Mises plasticity, Drucker plasticity, Hill's potential plasticity, crushable foam plasticity with volumetric hardening, crushable foam plasticity with isotropic hardening, ductile failure plasticity, rate-dependent Hill's potential plasticity, rate-dependent Mises plasticity, Drucker-Prager/Cap plasticity, and porous metal plasticity.

Problem description

This verification test consists of a set of single element models that include combinations of all of the available element types with all of the available material models. All of the elements are loaded with a tensile load defined by specifying the vertical velocity at the top nodes of each element with the bottom nodes fixed. One field variable, which increases from an initial value of 0 to a final value of 100, is defined at all of the nodes. Material properties are defined as a linear function of the field variable. For every material model only those element types available for the model are used. The undeformed meshes are shown in *Figure 1*, and the material properties are listed in *Table 1*.

Results and discussion

Figure 2 shows the history plot of Mises stress for the Mises plasticity model for all elements, except for pipe elements, which are consistent with beams. We can see the material softening because the yield stress drops as the field variable increases. *Figure 3* through *Figure 11* show the history plots of Mises stress for the other material models.

Input files

field_plastic.inp	Input data used in this analysis.
field_plastic_ef1.inp	External file referenced in this input.

Tables

Table 1: Material properties.

Material	Properties	fv=0	fv=100
Mises plasticity (density=8032)	E	193.1×10^9	160.1×10^9
	ν	0.3	0.3
	σ_0	206893	186893
	H	206893	186893
Drucker plasticity (density=1000)	E	2.1×10^7	1.9×10^7
	ν	0.3	0.3
	σ_0	40000	36000
	H	40000	39000
	β	40	39
	K	1.0	0.9
	ψ	20.0	19.0
Hill's plasticity (density=2500)	E	1.0×10^9	8.0×10^8
	ν	0.3	0.31
	σ_0	1.0×10^6	9.0×19^5
	H	4.0×10^5	3.7×10^5
Crushable foam with volumetric hardening (density=500)	E	5.0×10^6	3.0×10^6
	ν	0.3	0.0
	k	0.9	1.3
	k_t	0.1	0.1
Crushable foam with isotropic hardening (density=500)	E	5.0×10^6	3.0×10^6
	ν	0.3	0.0
	k	0.9	1.3
	ν_p	0.0	0.0
Ductile failure (density=5800)	E	2.0×10^8	1.8×10^8
	ν	0.3	0.3
	σ_0	2.0×10^5	1.8×10^5
	H	4.0×10^5	3.8×10^5
Hill's plasticity (density=5850)	E	1.8×10^8	2.0×10^8
(rate dependent)	ν	0.3	0.3
	σ_0	1.8×10^5	1.7×10^5
	H	−8000	−8000

Material	Properties	fv=0	fv=100
Mises plasticity (density=1500)	E	2.0×10^9	1.8×10^9
(rate dependent)	ν	0.4	0.4
	σ_0	6.0×10^7	5.5×10^7
	H	2.0×10^7	3.5×10^7
Drucker-Prager/Cap plasticity	E	30000	29000
(density=2.4×10^{-3})	ν	0.3	0.29
	d	100	99
	β	37.67	36.67
	R	0.1	0.11
	ϵ_{vol}^{pl}	0.0	0.0
	α	0.01	0.011
Porous metal plasticity	E	2.0×10^{11}	1.8×10^{11}
(density=7.7×10^7)	ν	0.33	0.33
	σ_0	7.5×10^8	7.5×10^8
	H	0.0	0.0

Figures

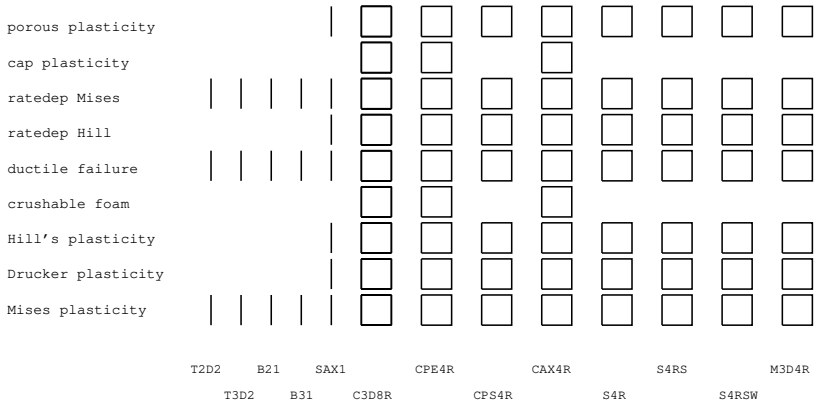


Figure 1: Field-variable-dependent material property test for inelastic materials.

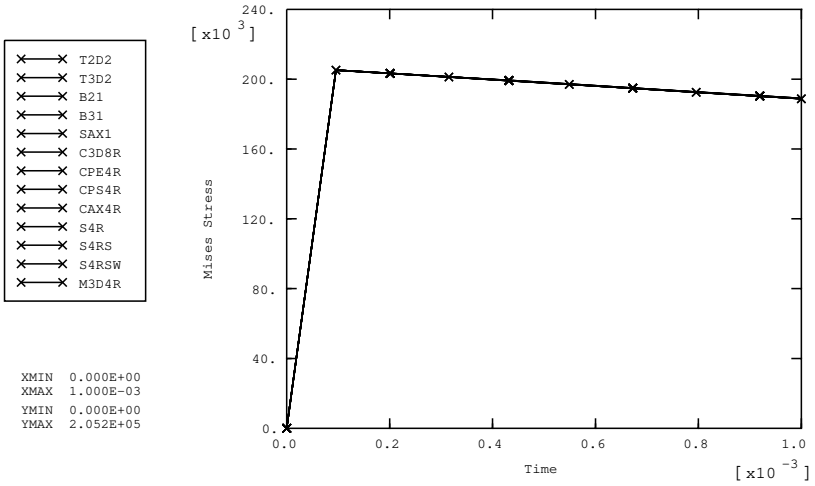


Figure 2: Mises stress versus time for Mises plasticity.

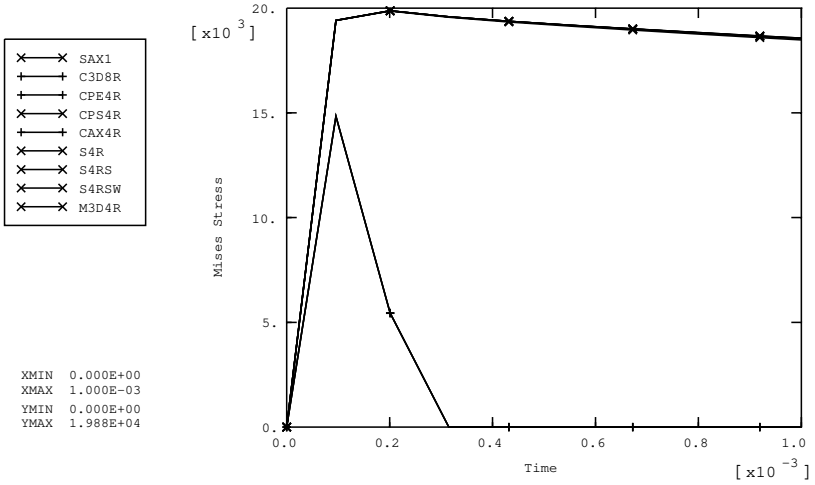


Figure 3: Mises stress versus time for Drucker plasticity.

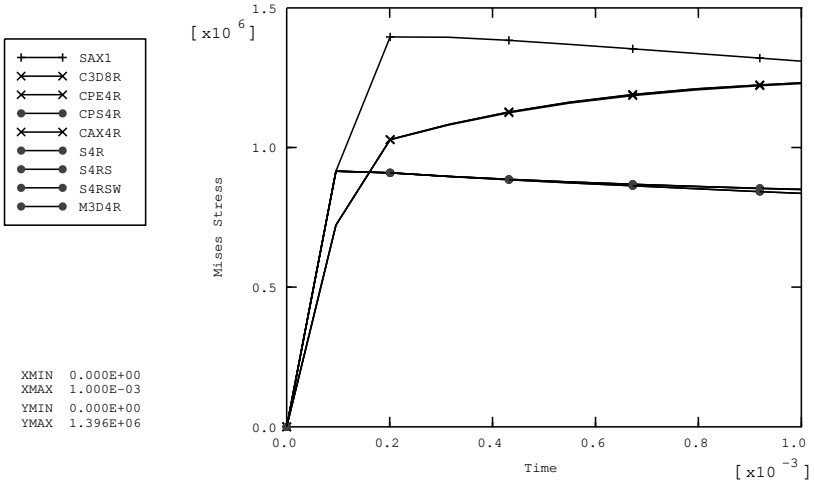


Figure 4: Mises stress versus time for Hill's plasticity.

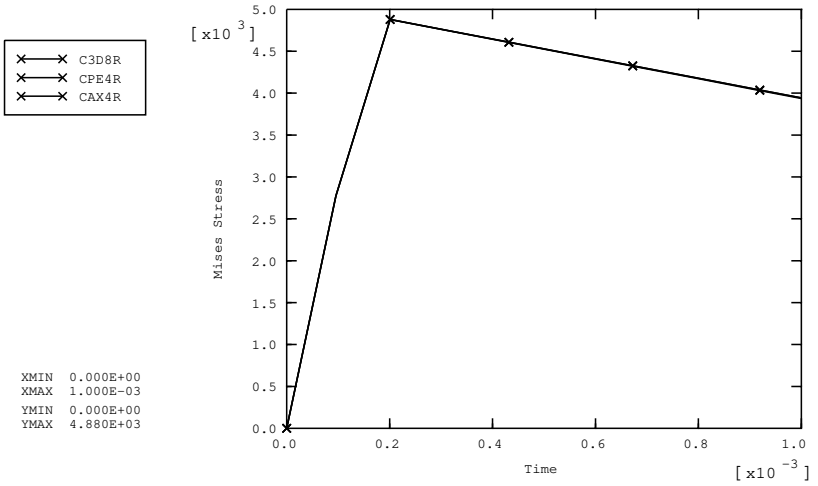


Figure 5: Mises stress versus time for crushable foam plasticity with volumetric hardening.

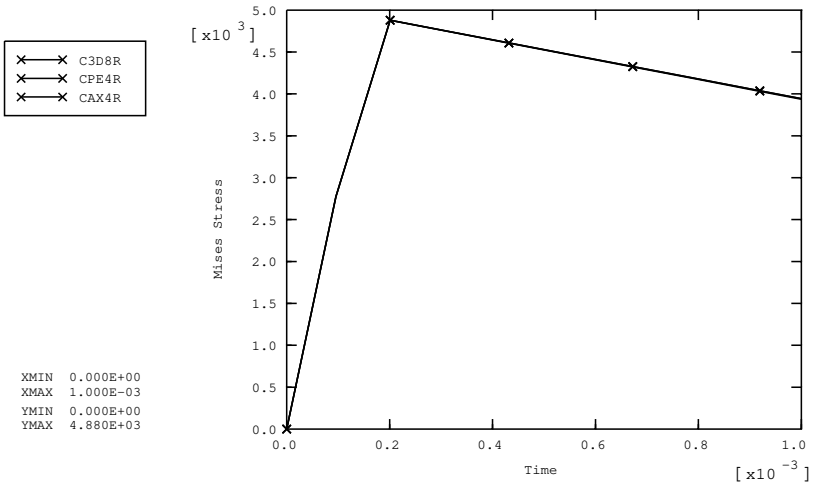


Figure 6: Mises stress versus time for crushable foam plasticity with isotropic hardening.

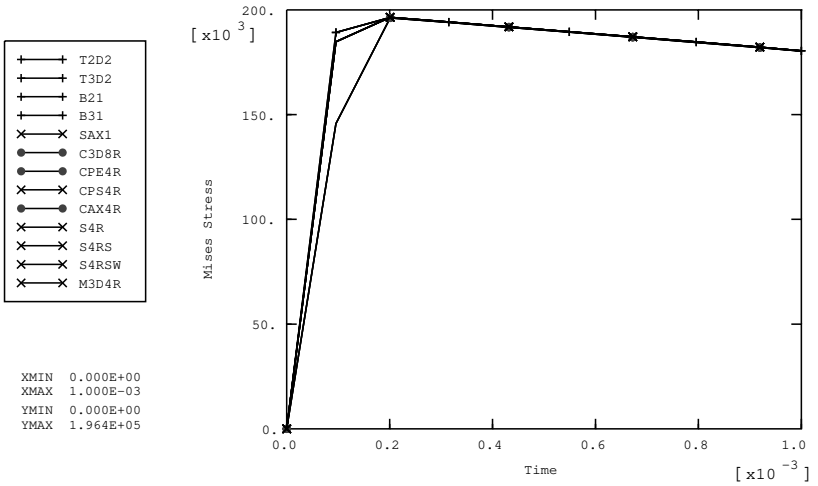


Figure 7: Mises stress versus time for ductile failure plasticity.

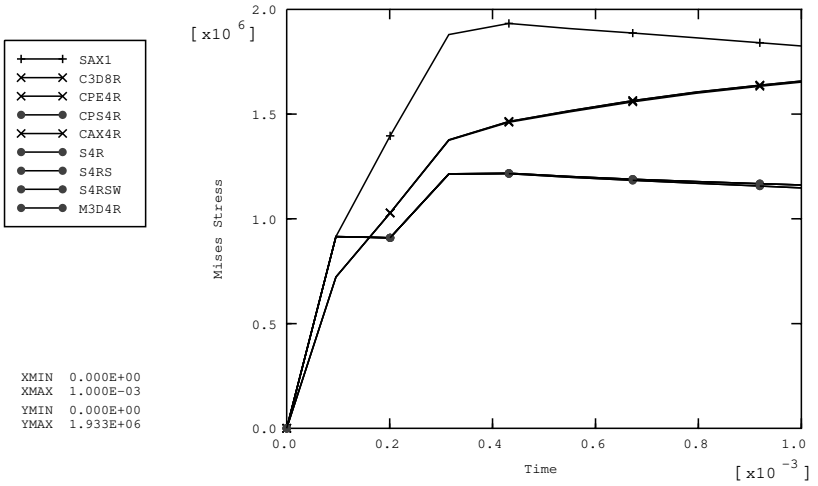


Figure 8: Mises stress versus time for rate-dependent Hill's plasticity.

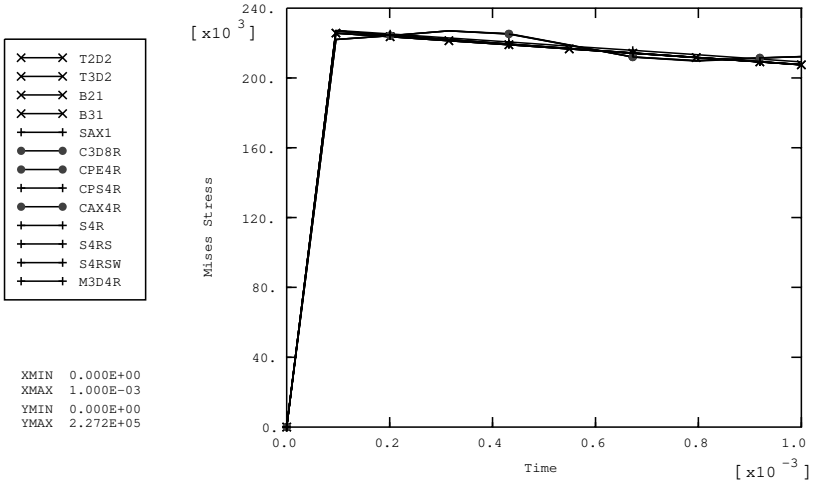


Figure 9: Mises stress versus time for rate-dependent Mises plasticity.

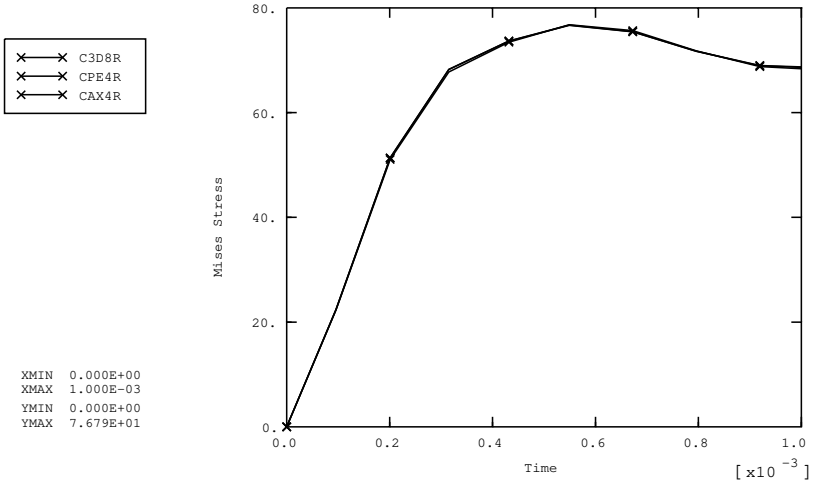


Figure 10: Mises stress versus time for Drucker-Prager/Cap plasticity.

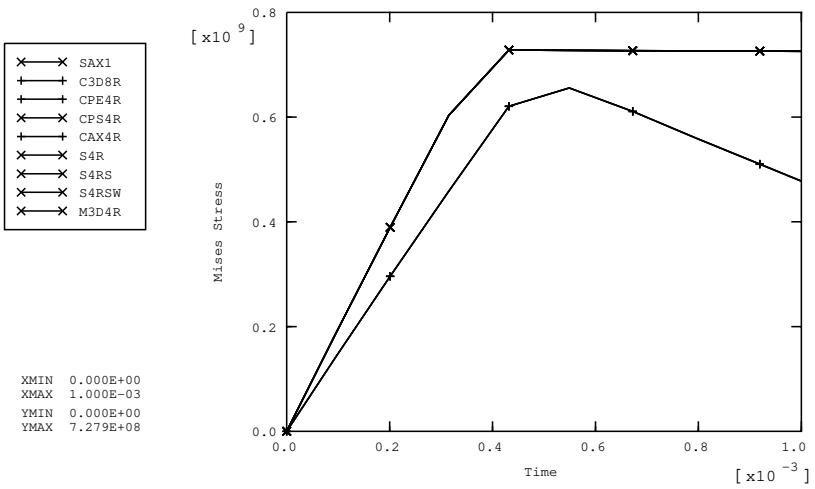


Figure 11: Mises stress versus time for porous metal plasticity.

Barlat anisotropic plasticity

Products: Abaqus/Standard Abaqus/Explicit

References:

- *Classical Metal Plasticity*
- *Nonquadratic Yield*

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D8R CPS4R

Features tested

Classical metal plasticity model with Barlat yield surfaces.

Problem description

This verification test contains a set of single-element models subject to uniaxial tension for various off-axis angles between the material directions and the loading direction. The test is designed to verify the yield stress ratios and strain ratios (Lankford's *r*-values) obtained in the simulations against the analytical solutions.

Model *Figure 1* shows the elements used in the analysis. All elements are subject to uniaxial tensile loading in the global *x*-direction. Each element has a unique material orientation. In the tests using C3D8R elements, the material orientation is assigned such that for the three rows of elements (from bottom to top), the loading direction lies in the material 1-2 plane, 2-3 plane, 3-1 plane, respectively; and for the seven columns of elements (from left to right), the off-axis angle of loading is 0°, 15°, 30°, 45°, 60°, 75°, and 90°, respectively. In the tests using CPS4R elements, the material orientation is assigned such that same off-axis angles of loading are tested.

Material

Elastic

Young's modulus, $E = 70.0 \times 10^3$

Poisson's ratio, $\upsilon = 0.33$

Plastic

Hardening:

Yield stress	Plastic strain
150.	0.000
265.	0.247

(The units are not important.)

Four materials with Barlat anisotropy coefficients are tested. Coefficients of the Barlat Yld2004-18p yield surface for material 2090-T3 and 6111-T4 are shown in *Table 1*. The yield surfaces of the material 2090-T3 predicted by the Barlat Yld2004-18p model

are illustrated in *Figure 2*, *Figure 3*, and *Figure 4* for three possible plane stress conditions, respectively. Material 2090-T3 is also modeled with Barlat Yld2000-2d yield surfaces, and the coefficients are listed in *Table 3*. The yield surface of the material 2090-T3 predicted by the Barlat Yld2000-2d model is shown in *Figure 8*. The other two materials, 2024-T3 and 2008-T4, use the Barlat Yld91 yield surface with anisotropy coefficients as shown in *Table 2*.

In addition, the Mises isotropic plasticity is recovered with an equivalent definition for the Barlat yield surfaces.

Results and discussion

The yield stress ratio is defined as

$$R = \frac{\sigma_x}{\sigma^0},$$

where σ_x is the stress in the loading direction that can be obtained as output variable S11 in the global frame and σ^0 is the reference yield stress that can be obtained as output variable YIELDPOT.

The strain ratio (Lankford's *r*-value) is defined as

$$r = \frac{\varepsilon_y}{\varepsilon_z},$$

where ε_y and ε_z are the plastic strains in the width and thickness directions, respectively, corresponding to output variables PE22 and PE33 in the global frame.

The results of the yield stress and strain ratios agree well with the analytical solutions for the four materials. Comparisons of the simulation results using the Barlat Yld2004-18p model and analytical solutions for the material 2090-T3 with loading applied on three material planes are shown in *Figure 5*, *Figure 6*, and *Figure 7*. *Figure 9* compares the plastic strains of material 2090-T3 at different off-axis loading angles in the 1-2 plane. The predictions given by the Barlat Yld2004-18p model and the Barlat Yld2000-2d model match very well.

The yield stress ratio and strain ratio of the Mises plasticity are always one. The models with equivalent definitions of the Mises yield surface also match the analytical solutions.

Input files

Abaqus/Standard input files

barlat_c3d8r_2090t3_std.inp	Analysis using C3D8R elements and the Barlat Yld2004-18p yield surface for the material 2090-T3.
barlat_cps4r_2090t3_std.inp	Analysis using CPS4R elements and the Barlat Yld2004-18p yield surface for the material 2090-T3.
barlat2000_c3d8r_2090t3_std.inp	Analysis using C3D8R elements and the Barlat Yld2000-2d yield surface for the material 2090-T3.
barlat2000_cps4r_2090t3_std.inp	Analysis using CPS4R elements and the Barlat Yld2000-2d yield surface for the material 2090-T3.
barlat_c3d8r_6111t4_std.inp	Analysis using C3D8R elements and the Barlat Yld2004-18p yield surface for the material 6111-T4.
barlat_cps4r_6111t4_std.inp	Analysis using CPS4R elements and the Barlat Yld2004-18p yield surface for the material 6111-T4.

barlat_c3d8r_mises_std.inp	Analysis using C3D8R elements and the Barlat Yld2004-18p yield surface with an equivalent definition of the Mises yield surface.
barlat_cps4r_mises_std.inp	Analysis using CPS4R elements and the Barlat Yld2004-18p yield surface with an equivalent definition of the Mises yield surface.
barlat91_c3d8r_2024t3_std.inp	Analysis using C3D8R elements and the Barlat Yld91 yield surface for the material 2024-T3.
barlat91_cps4r_2024t3_std.inp	Analysis using CPS4R elements and the Barlat Yld91 yield surface for the material 2024-T3.
barlat91_c3d8r_2008t4_std.inp	Analysis using C3D8R elements and the Barlat Yld91 yield surface for the material 2008-T4.
barlat91_cps4r_2008t4_std.inp	Analysis using CPS4R elements and the Barlat Yld91 yield surface for the material 2008-T4.
barlat91_c3d8r_mises_std.inp	Analysis using C3D8R elements and the Barlat Yld91 yield surface with an equivalent definition of the Mises yield surface.
barlat91_cps4r_mises_std.inp	Analysis using CPS4R elements and the Barlat Yld91 yield surface with an equivalent definition of the Mises yield surface.

Abaqus/Explicit input files

barlat_c3d8r_2090t3.inp	Analysis using C3D8R elements and the Barlat Yld2004-18p yield surface for the material 2090-T3.
barlat_cps4r_2090t3.inp	Analysis using CPS4R elements and the Barlat Yld2004-18p yield surface for the material 2090-T3.
barlat2000_c3d8r_2090t3.inp	Analysis using C3D8R elements and the Barlat Yld2000-2d yield surface for the material 2090-T3.
barlat2000_cps4r_2090t3.inp	Analysis using CPS4R elements and the Barlat Yld2000-2d yield surface for the material 2090-T3.
barlat_c3d8r_6111t4.inp	Analysis using C3D8R elements and the Barlat Yld2004-18p yield surface for the material 6111-T4.
barlat_cps4r_6111t4.inp	Analysis using CPS4R elements and the Barlat Yld2004-18p yield surface for the material 6111-T4.
barlat_c3d8r_mises.inp	Analysis using C3D8R elements and the Barlat Yld2004-18p yield surface with an equivalent definition of the Mises yield surface.
barlat_cps4r_mises.inp	Analysis using CPS4R elements and the Barlat Yld2004-18p yield surface with an equivalent definition of the Mises yield surface.
barlat91_c3d8r_2024t3.inp	Analysis using C3D8R elements and the Barlat Yld91 yield surface for the material 2024-T3.
barlat91_cps4r_2024t3.inp	Analysis using CPS4R elements and the Barlat Yld91 yield surface for the material 2024-T3.
barlat91_c3d8r_2008t4.inp	Analysis using C3D8R elements and the Barlat Yld91 yield surface for the material 2008-T4.
barlat91_cps4r_2008t4.inp	Analysis using CPS4R elements and the Barlat Yld91 yield surface for the material 2008-T4.
barlat91_c3d8r_mises.inp	Analysis using C3D8R elements and the Barlat Yld91 yield surface with an equivalent definition of the Mises yield surface.
barlat91_cps4r_mises.inp	Analysis using CPS4R elements and the Barlat Yld91 yield surface with an equivalent definition of the Mises yield surface.

References

- Barlat, F., H. Aretz, J. W. Yoon, M. E. Karabin, J. C. Brem, and R. E. Dick, "Linear Transformation-based Anisotropic Yield Functions," *International Journal of Plasticity*, vol. 21, pp. 1009–1039, 2005.
- Barlat, F., D. J. Lege, and J. C. Brem, "A Six-Component Yield Function for Anisotropic Materials," *International Journal of Plasticity*, vol. 7, pp. 693–712, 1991.
- Barlat, F., J. C. Brem, J. W. Yoon, K. Chung, R. E. Dick, D. J. Lege, F. Pourboghrat, S. H. Choi, and E. Chu, "Plane Stress Yield Function for Aluminum Alloy Sheets—Part 1: Theory," *International Journal of Plasticity*, vol. 19, pp. 1297–1319, 2003.

Tables

Table 1: Anisotropic coefficients of the Barlat Yld2004-18p yield surface for material 6111-T4 and 2090-T3 (*Barlat et. al, 2005*).

Abaqus notation	Barlat's notation	Material 2090-T3	Material 6111-T4
α	a	8.0	8.0
c'_{1122}	c'_{12}	-0.069888	1.078271
c'_{1133}	c'_{13}	0.936408	1.078271
c'_{2211}	c'_{21}	0.079143	1.216463
c'_{2233}	c'_{23}	1.003060	1.223867
c'_{3311}	c'_{31}	0.524741	1.093105
c'_{3322}	c'_{32}	1.363180	0.889161
c'_{1212}	$\frac{1}{2}c'_{66}$	0.477161	0.674547
c'_{1313}	$\frac{1}{2}c'_{55}$	0.534530	0.278587
c'_{2323}	$\frac{1}{2}c'_{44}$	0.511885	0.250955
c''_{1122}	c''_{12}	0.981171	0.775366
c''_{1133}	c''_{13}	0.476741	0.922743
c''_{2211}	c''_{21}	0.575316	0.765487
c''_{2233}	c''_{23}	0.866827	0.793356
c''_{3311}	c''_{31}	1.145010	0.918689
c''_{3322}	c''_{32}	-0.079294	1.027625
c''_{1212}	$\frac{1}{2}c''_{66}$	0.702310	0.294894

Abaqus notation	Barlat's notation	Material 2090-T3	Material 6111-T4
c''_{1313}	$\frac{1}{2}c''_{55}$	0.573550	0.556137
c''_{2323}	$\frac{1}{2}c''_{44}$	0.525830	0.557917

Table 2: Anisotropic coefficients of the Barlat Yld91 yield surface for material 2008-T4 and 2024-T3 (*Barlat et. al, 1991*).

Coefficients	Material 2024-T3	Material 2008-T4
α	8.0	11.0
$\overline{a}$	1.378	1.222
$\overline{b}$	1.044	1.013
$\overline{c}$	0.955	0.985
$\overline{f}$	1.0	1.0
$\overline{g}$	1.0	1.0
$\overline{h}$	1.210	1.0

Table 3: Anisotropic coefficients of the Barlat Yld2000-2d yield surface for material 2090-T3 (*Barlat et. al, 2003*).

Coefficients	Material 2090-T3
α	8
α_1	0.4865
α_2	1.378
α_3	0.7536
α_4	1.0246
α_5	1.0363
α_6	0.9036
α_7	1.2321
α_8	1.4858

Figures

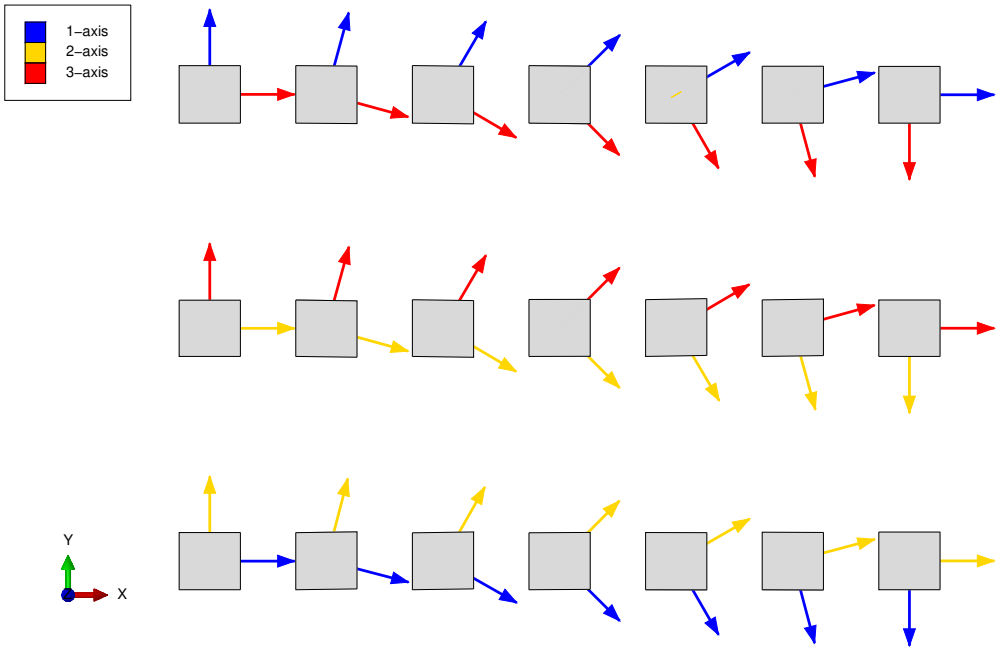


Figure 1: Elements and their material orientations.

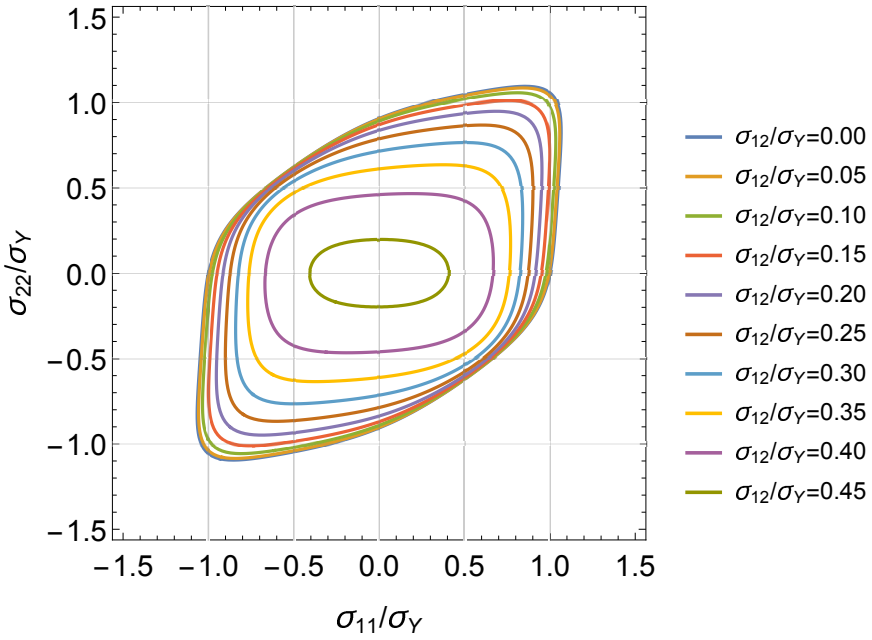


Figure 2: Yield surface of material 2090-T3 with an assumption of plane stress in the material 1–2 plane.

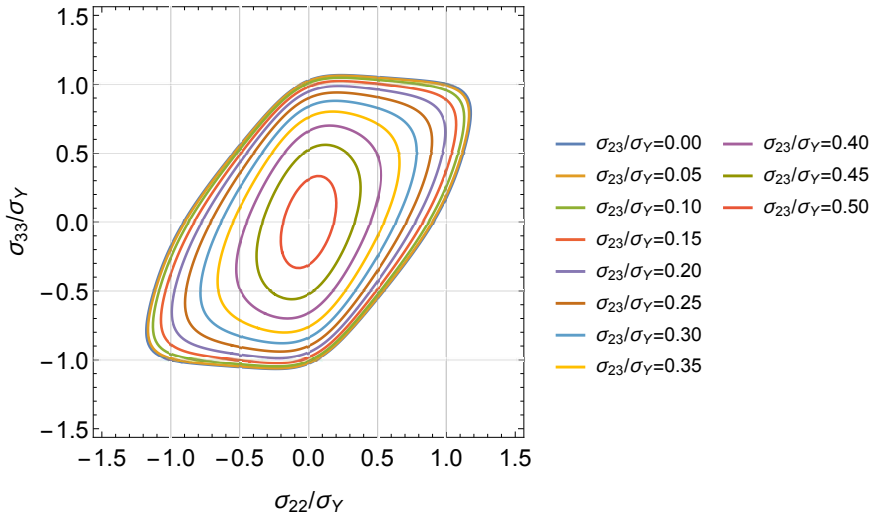


Figure 3: Yield surface of material 2090-T3 with an assumption of plane stress in the material 2–3 plane.

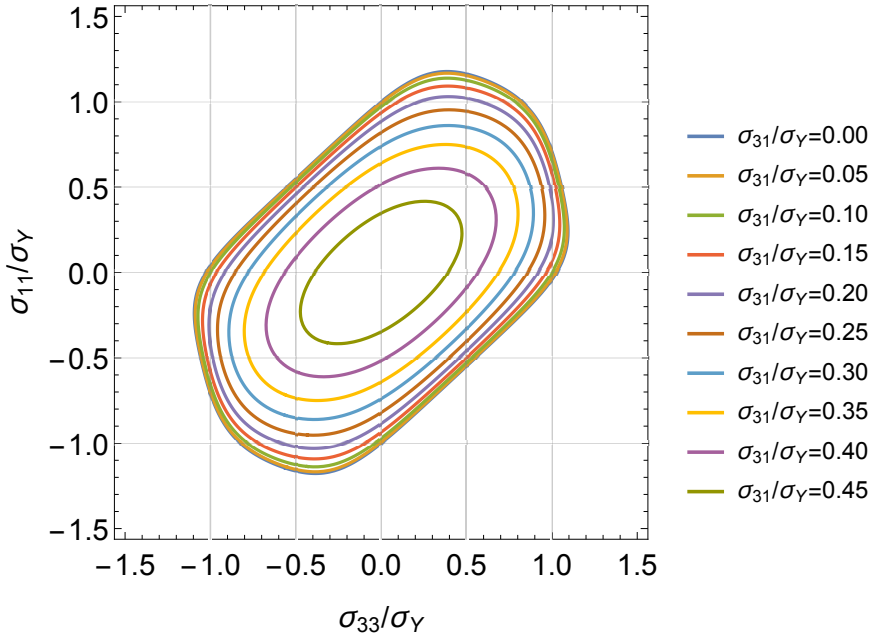


Figure 4: Yield surface of material 2090-T3 with an assumption of plane stress in the material 3–1 plane.

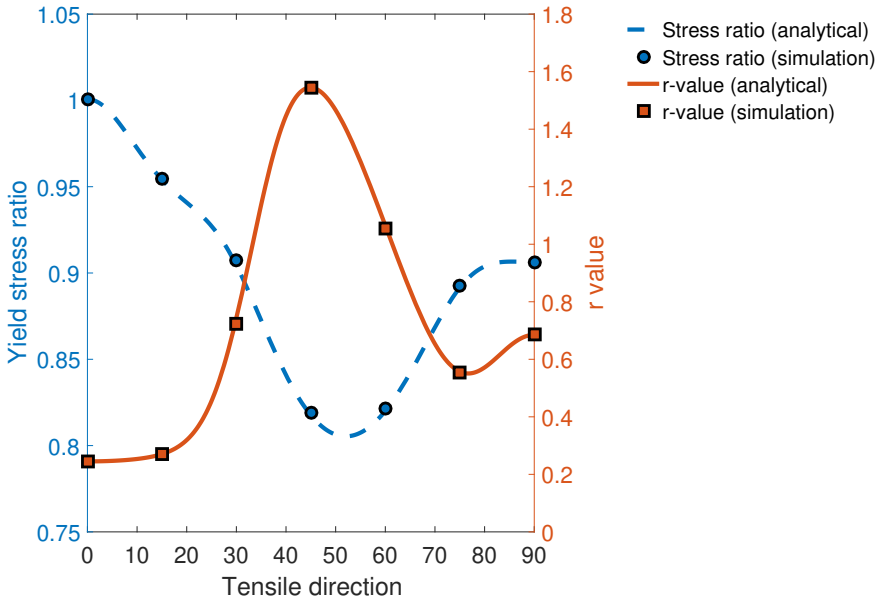


Figure 5: Yield stress ratios and strain ratios (r-values) of off-axis uniaxial loading in the material 1–2 plane for material 2090-T3.

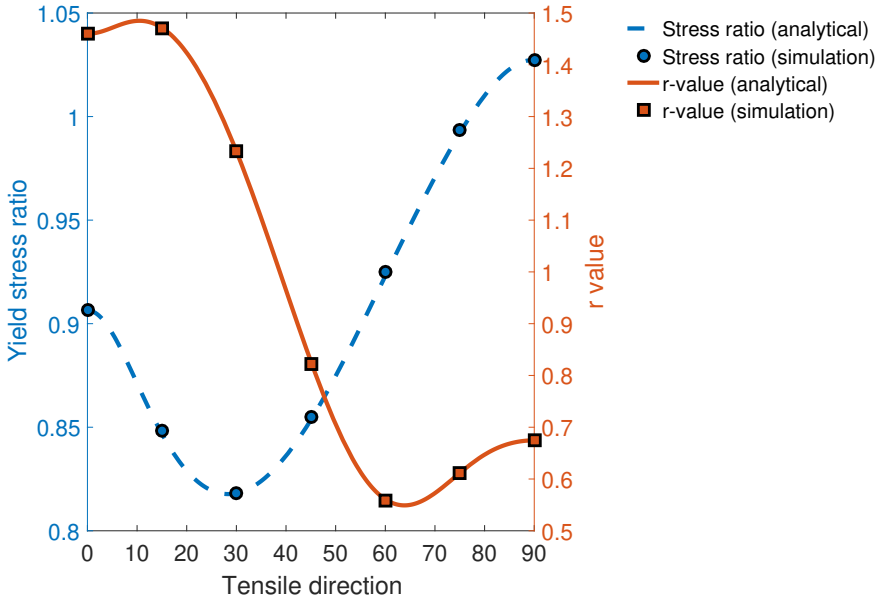


Figure 6: Yield stress ratios and strain ratios (r-values) of off-axis uniaxial loading in the material 2–3 plane for material 2090-T3.

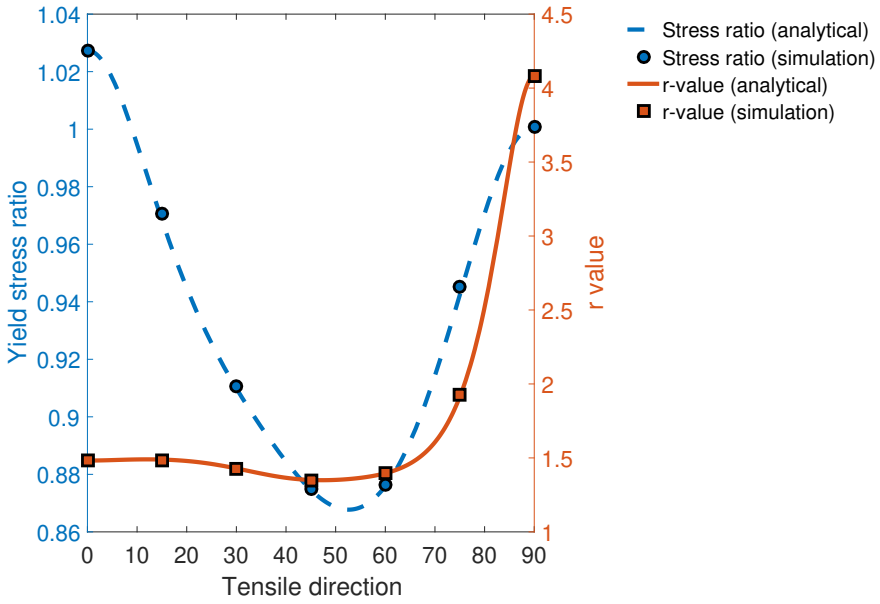


Figure 7: Yield stress ratios and strain ratios (r-values) of off-axis uniaxial loading in the material 3–1 plane for material 2090-T3.

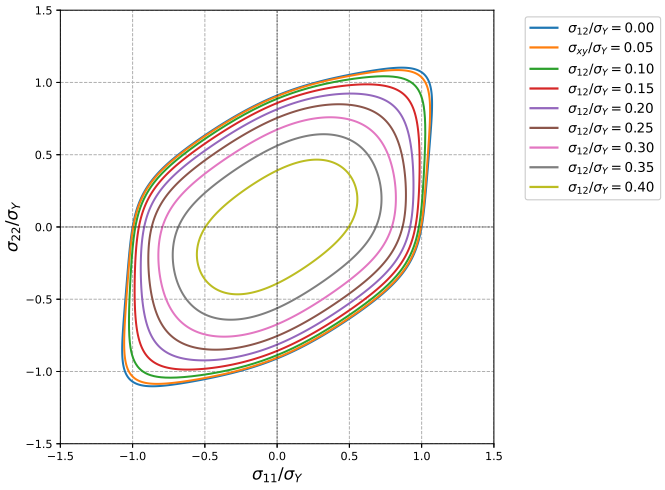


Figure 8: Yield surface of material 2090-T3 predicted with the Barlat Yld2000-2d model.

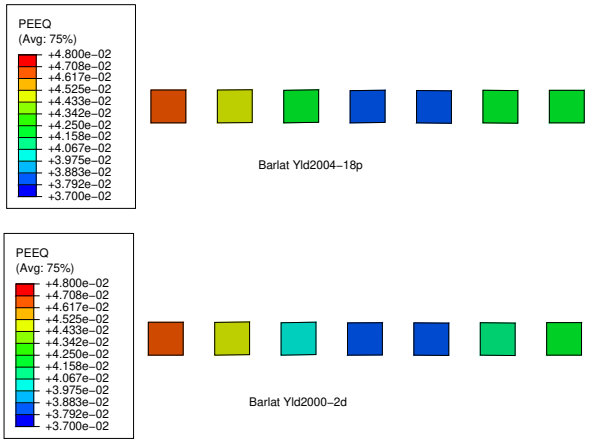


Figure 9: Comparisons of the plastic strains predicted by Barlat Yld2004-18p and Barlat Yld2000-2d models.

Johnson-Cook plasticity

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

T2D2 T3D2 B21 B31 SAX1 C3D8R CPE4R CPS4R CAX4R S4R S4RS S4RSW M3D4R

Features tested

Johnson-Cook plasticity model.

Problem description

This verification problem tests single-element models that are run under simple loading conditions (uniaxial tension, uniaxial compression, and simple shear). The purpose of this example is to test the Johnson-Cook plasticity model by comparing it to the Mises plasticity model with equivalent plastic hardening. *Figure 1* shows the 26 elements used in the analysis in their original shapes. The elements in the top row are modeled using the Johnson-Cook material model; the elements in the bottom row are modeled using the Mises plasticity model with an equivalent hardening curve. The elastic material properties are Young's modulus = 124 GPa and Poisson's ratio = 0.34. The plastic hardening is chosen to be

$$\overline{\sigma} = 90 + 292\left(\overline{\varepsilon}^{pl}\right)^{0.31},$$

where $\overline{\sigma}$ is the yield stress (unit in MPa) and $\overline{\varepsilon}^{pl}$ is the equivalent plastic strain. The material properties are those of OFHC copper as reported by *Johnson and Cook (1985)*. A plot of $\overline{\sigma}$ versus $\overline{\varepsilon}^{pl}$ is shown in *Figure 2*.

Results and discussion

Results obtained by using the Johnson-Cook material model are smoother than corresponding results obtained by using the Mises plasticity model. The Johnson-Cook model has an analytical expression for the nonlinear hardening curve; whereas the hardening curve for Mises plasticity model is linear between input data points, as shown in *Figure 2*. *Figure 3* shows the comparison of the stress-strain responses obtained with the Johnson-Cook and the Mises plasticity models using the C3D8R element under uniaxial tension; *Figure 4* shows the comparison of the stress-strain responses obtained with the Johnson-Cook and the Mises plasticity models using the CPE4R element under uniaxial compression; *Figure 5* shows the comparison of the stress-strain responses obtained with the Johnson-Cook and the Mises plasticity models using the CPE4R element under simple shear.

Input files

Abaqus/Standard input files

johnsoncook_s.inp	Uniaxial tension test.
johnsoncookinit_pre_s.inp	Uniaxial compression test, nonzero initial conditions for $\overline{\varepsilon}^{pl}$.

Abaqus/Explicit input files

johnsoncook.inp	Uniaxial tension test.
johnsoncook_pre.inp	Uniaxial compression test.
johnsoncook_shr.inp	Simple shear test.
johnsoncookinit.inp	Uniaxial tension test, nonzero initial conditions for $\bar{\varepsilon}^{pl}$.
johnsoncookinit_pre.inp	Uniaxial compression test, nonzero initial conditions for $\bar{\varepsilon}^{pl}$.
johnsoncookinit_shr.inp	Simple shear test, nonzero initial conditions for $\bar{\varepsilon}^{pl}$.

References

- Johnson, G. R., and W. H. Cook, “Fracture Characteristics of Three Metals Subjected to Various Strains, Strain rates, Temperatures and Pressures,” Engineering Fracture Mechanics, vol. 21, no. 1, pp. 31–48, 1985.

Figures

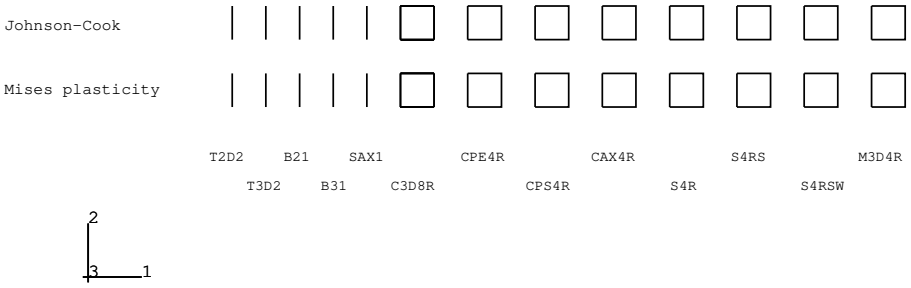


Figure 1: Johnson-Cook plasticity test cases.

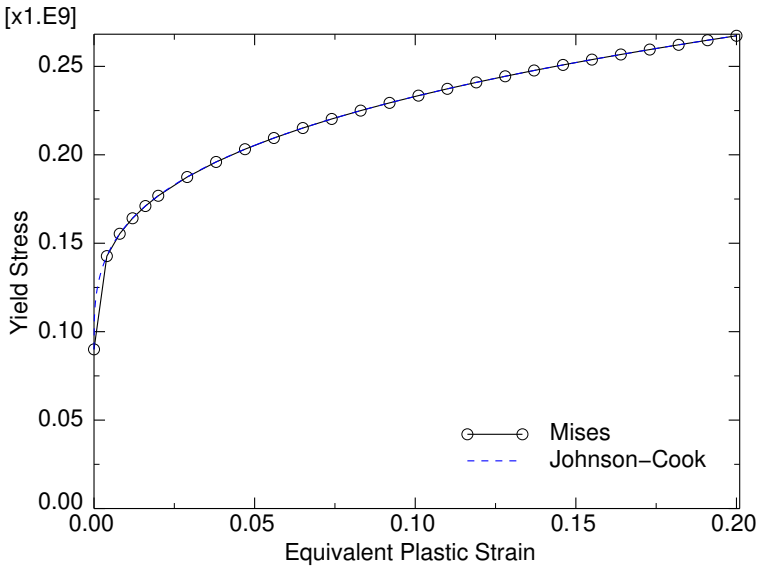


Figure 2: Hardening curve: yield stress versus equivalent plastic strain.

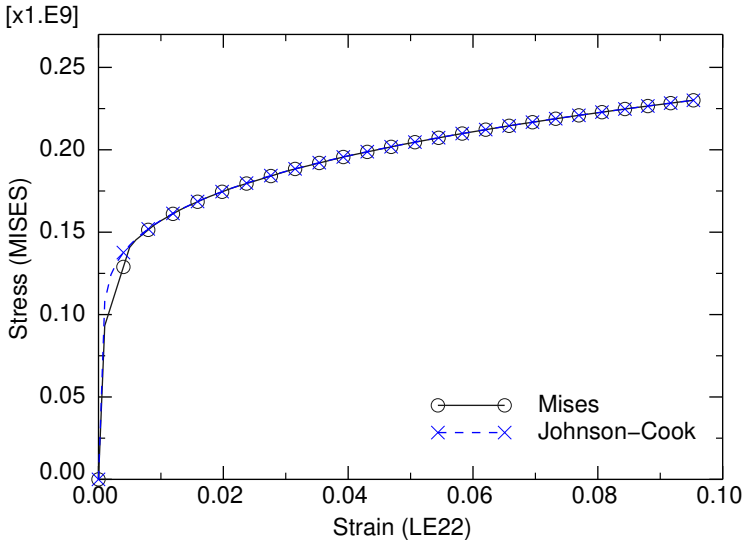


Figure 3: Uniaxial tension comparison (C3D8R element).

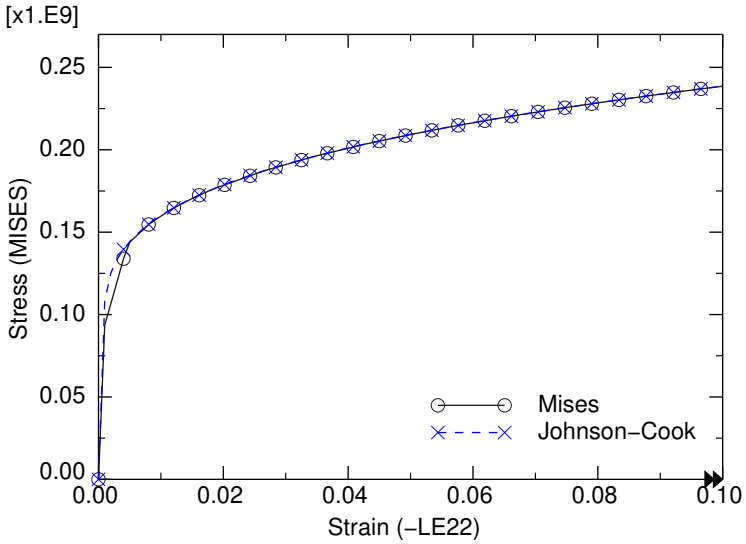


Figure 4: Uniaxial compression comparison (CPE4R element).

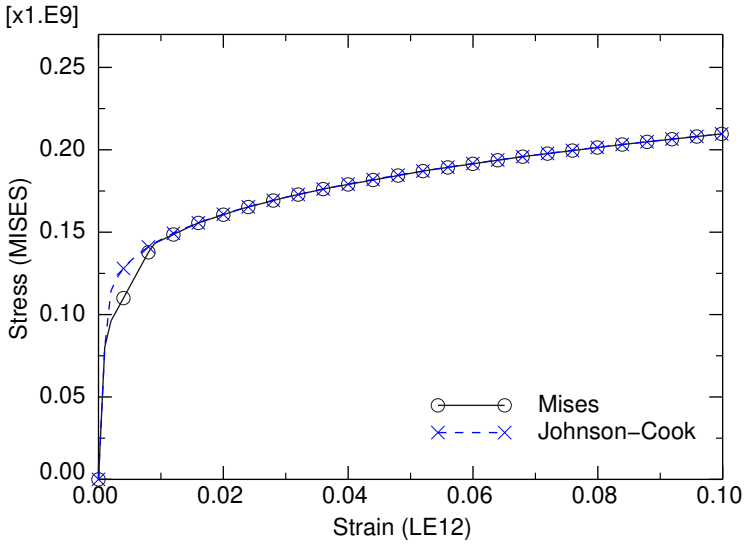


Figure 5: Simple shear comparison (CPE4R element).

Porous metal plasticity

Products: Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D8R CPE4R CPS4R

Features tested

Porous plasticity model.

Problem description

This problem contains 16 one-element verification problems that are all run in one input file. The purpose of this example is to test the porous plasticity model. Three different element types are tested (C3D8R, CPE4R, CPS4R). *Figure 1* shows the 16 elements used in the analysis in their original and deformed shapes. The dashed lines represent the original mesh. The 8-node brick element (C3D8R) appears twice in each row: in one case boundary conditions are applied to constrain the out-of-plane displacement so that the C3D8R element simulates plane strain conditions, and in the second case no out-of-plane displacement boundary conditions are specified so that the C3D8R element simulates plane stress conditions. The original length of each side of the elements is 1.

This example problem is designed to test the following features:

- plane strain, plane stress, and three-dimensional cases
- tension, compression, and simple shear deformations
- void nucleation and void growth

This is accomplished as described below.

The loading on the elements in row (a) represents uniaxial tensile loading in the x-direction.

In rows (b) and (c) in *Figure 1* the left and right, top and bottom nodes of each element are given equal and opposite prescribed constant velocities in the x- and y-directions to generate biaxial compressive and tensile loading for both plane strain and plane stress cases.

In row (d) in *Figure 1* the bottom and top nodes of each element are given equal and opposite prescribed constant velocities in the x-direction to generate a simple shear loading.

The elements in rows (b), (c), and (d) of *Figure 1* are assigned material definitions with no void nucleation and with the coefficients $q_1 = q_2 = q_3 = 1.0$. The behavior of the matrix material is assumed to be perfectly plastic with a yield stress $\sigma_0 = 1.0$. The elements in row (a) of *Figure 1* are assigned material definitions with void nucleation ($\varepsilon_N = 0.3$, $s_N = 0.1$, $f_N = 0.04$) and with coefficients $q_1 = 1.5$, $q_2 = 1.0$, $q_3 = 2.25$. The behavior for this matrix material includes hardening. The elastic properties are $E = 300$, $\nu = 0.3$, and a density of 0.001 is used in both material definitions. The initial relative density is assumed to be 0.95 in all cases.

Results and discussion

The results obtained from the plane strain and plane stress elements in all the tests are identical to the corresponding results obtained from the three-dimensional elements where plane strain and plane stress boundary conditions are applied. The names of the individual curves that appear in the graph legend are a concatenation of the output variable names, an underscore (_), and a number. The number refers to the element number. For example, PEEQ–Q_1 refers to the Mises stress versus equivalent plastic strain curve for element 1.

Figure 2 shows the variation of the volume fraction of voids as a function of time. The figure indicates that the void volume fraction remains constant during pure shear (line 1). In the compression test the void volume fraction reduces as the pressure increases (lines 2 and 3). Once the voids are fully closed, the material becomes almost incompressible. In the multiaxial and uniaxial tension tests the voids grow (lines 4 through 7) and new voids may nucleate (lines 6 and 7) for the material where void nucleation is specified.

Figure 3 and *Figure 4* show the variation of Mises stress with pressure stress and the variation of Mises stress with equivalent plastic strain. The evolution of the stress path of a material point is depicted through these figures. The influence of void closure and void growth on the pressure stress is shown in *Figure 5*. The figure contains the results obtained from the plane strain biaxial compression (line 1) and tension (line 2) tests. In the compression test the response is elastic, followed by plastic hardening until voids are closed, which is finally followed by incompressible behavior. In tension elastic behavior is followed by softening as voids grow.

The results that are obtained with Abaqus/Explicit are the same as those obtained in Abaqus/Standard.

Input files

gurson.inp	Input data used in this analysis.
gurson_mod.inp	Identical to gurson.inp except that the initial relative density is specified using **INITIAL CONDITIONS*, TYPE=RELATIVE DENSITY.

Figures

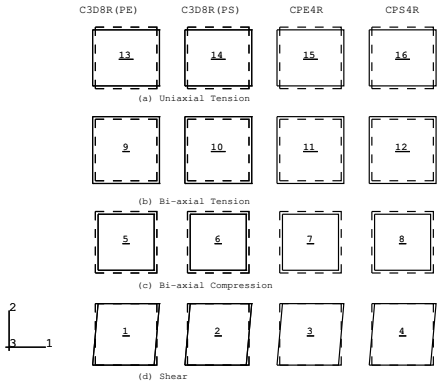


Figure 1: Deformed shape for one-element porous plasticity tests.

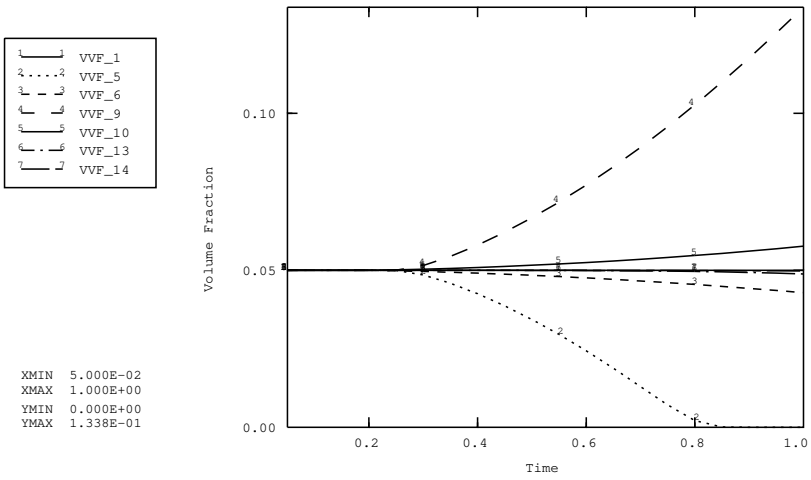


Figure 2: Volume fraction of voids versus time.

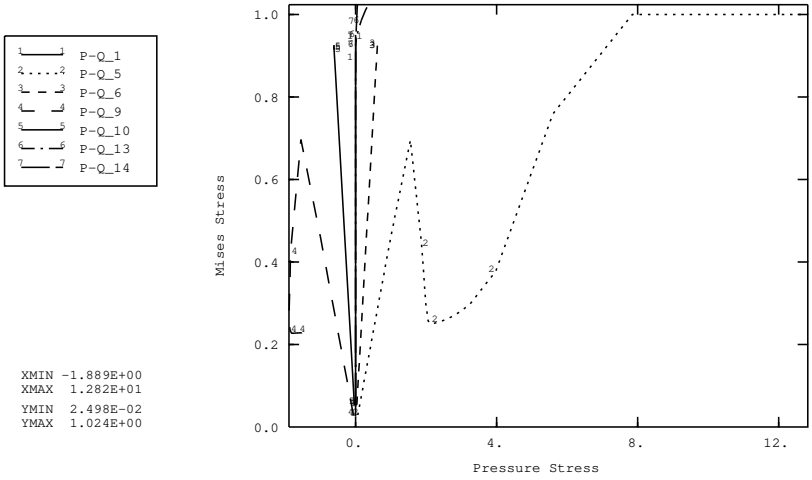


Figure 3: Evolution of stress state in p–q space.

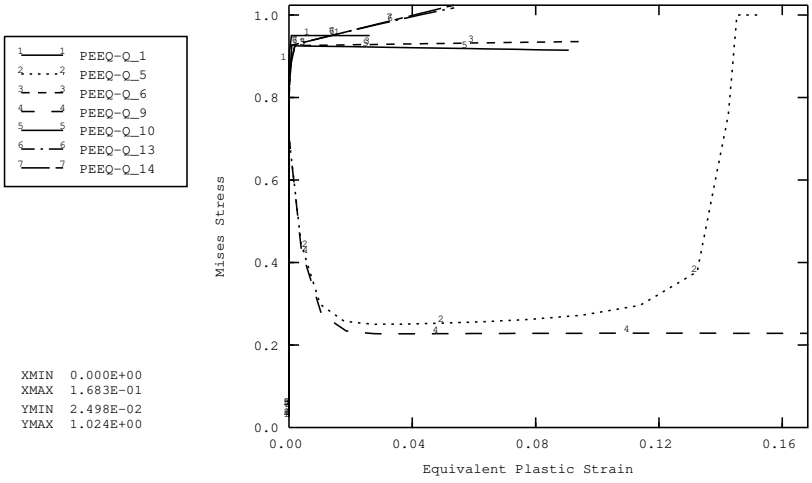


Figure 4: Mises stress versus equivalent plastic strain.

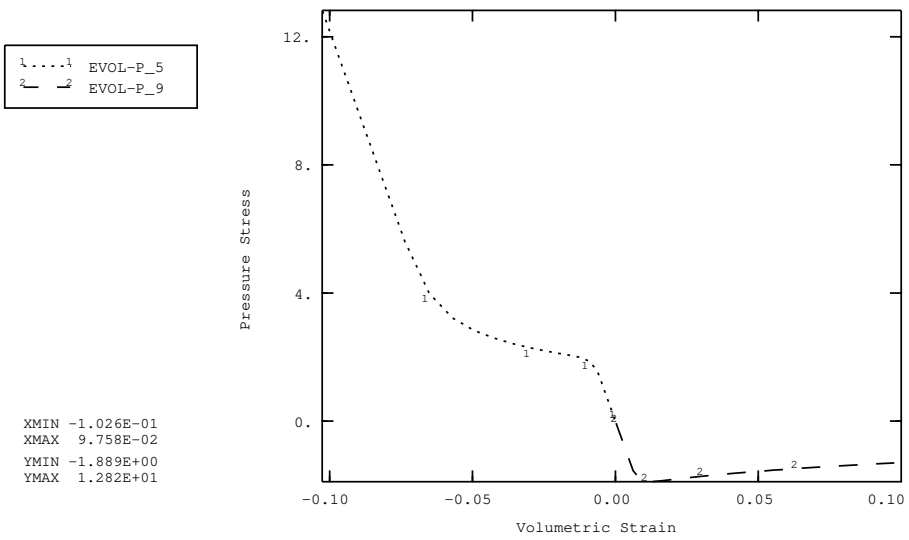


Figure 5: Pressure stress versus volume strain.

Drucker-Prager plasticity

Products: Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D8R CPE4R CPS4R

Features tested

Extended Drucker-Prager plasticity with third stress invariant.

Problem description

This problem contains 16 one-element verification problems that are all run in one input file. The problem exercises the extended Drucker-Prager plasticity material model using associated and nonassociated flow rules. Three different element types are tested (C3D8R, CPE4R, CPS4R). *Figure 1* shows the 16 elements used in the analysis in their original and deformed shapes. The dashed lines illustrate the original mesh. The 8-node brick element (C3D8R) appears twice in each row: in one case boundary conditions are applied to constrain the out-of-plane displacement so that the C3D8R element generates plane strain results, in the second case no out-of-plane displacement boundary conditions are specified so that the C3D8R element generates plane stress results. The original length of each side of the elements is $L_0 = 1$.

This example problem is designed to test the following features:

- plane strain, plane stress, and three-dimensional cases
- compression and simple shear deformations
- associated and nonassociated flow
- third invariant dependence (the value of K)

This is accomplished as described below.

In rows (a) and (c) in *Figure 1* the bottom and top nodes of each element are given equal and opposite prescribed constant velocities in the y-direction to generate a compressive loading. For rows (b) and (d) in *Figure 1* the top nodes of each element are given prescribed velocities in the x-direction, while the bottom nodes are fixed, thereby leading to simple shear modes.

The elements in rows (a) and (c) in *Figure 1* are assigned material definitions with an associated flow rule: the friction angle (β) and dilation angle (ψ) are each 40°. The elements in rows (b) and (d) in *Figure 1* are assigned material definitions with a nonassociated flow rule: the friction angle (β) is 40°, while the dilation angle (ψ) is 20°.

The value of K is a measure of the strength differences in tension and compression. The yield surface will not be circular in the deviatoric stress space if K does not have a value of 1.0. In the first two rows of elements in *Figure 1* a value of $K = 0.8$ is assigned. In rows (c) and (d) in *Figure 1* a value of $K = 1.0$ is assigned to the elements.

The plastic stress-strain relationship is defined using the extended Drucker-Prager plasticity model with hardening. Perfect plasticity is assumed, with the yield stress in uniaxial compression $\sigma_0 = 40 \times 10^3$. The elastic properties are $E = 20 \times 10^6$, $\nu = 0.3$. Material densities are 1000.

Results and discussion

Figure 2 shows the plot of Mises stresses versus pressure in the plane strain cases (elements C3D8 and CPE4R) with an associated flow rule. This demonstrates the pressure-dependent nature of the material. In the case of $K = 1.0$, the slope of the curve corresponds to the tangent of 40°.

Figure 3 shows the plot of Mises stresses against pressure in the plane stress cases (elements C3D8 and CPS4R) with an associated flow rule.

Figure 4 shows the plot of Mises stresses versus pressure in the plane strain cases (elements C3D8 and CPE4R) with a nonassociated flow rule. In the case of $K = 1.0$, the slope of the curve corresponds to the tangent of 40°.

Figure 5 shows the plot of Mises stresses against pressure in the plane stress cases (elements C3D8 and CPS4R) with a nonassociated flow rule.

When K is less than 1.0, the slope of the Mises stress versus pressure curve will be less than or equal to the frictional angle. This depends on the plastic strain path in the noncircular deviatoric space.

Figure 6 contains eight curves of the time history response of equivalent plastic strain for the eight elements that have a value of $K = 0.8$. *Figure 7* contains eight history curves of equivalent plastic strain for the eight elements that have a value of $K = 1.0$. Only four curves are visible in *Figure 6* and *Figure 7* because the three-dimensional results for the C3D8R elements reproduce the plane strain and plane stress results. As discussed above, boundary conditions were applied to the C3D8R elements to achieve this correspondence. This serves as a check that both the two- and three-dimensional material models achieve the same results.

The results that are obtained with Abaqus/Explicit are the same as those obtained with Abaqus/Standard.

Input files

drucker.inp	Input data used in this analysis.

Figures

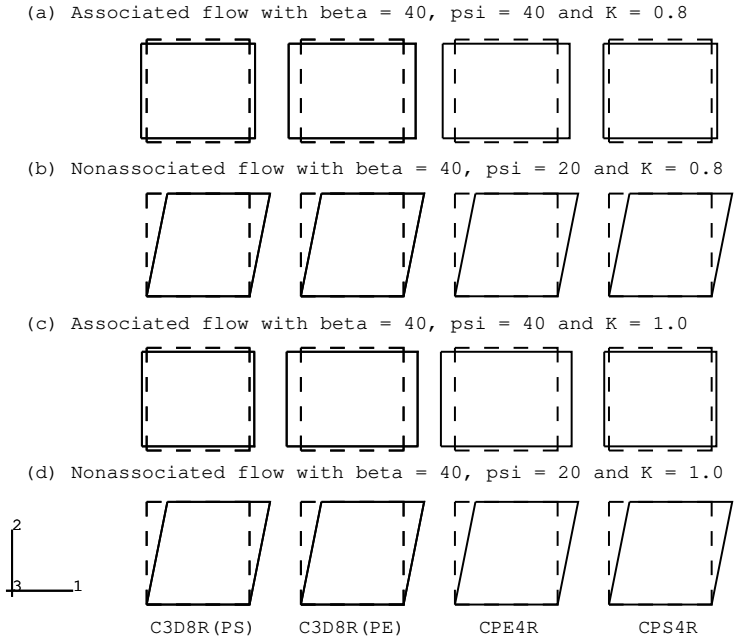


Figure 1: Deformed shape for one-element Drucker-Prager plasticity tests.

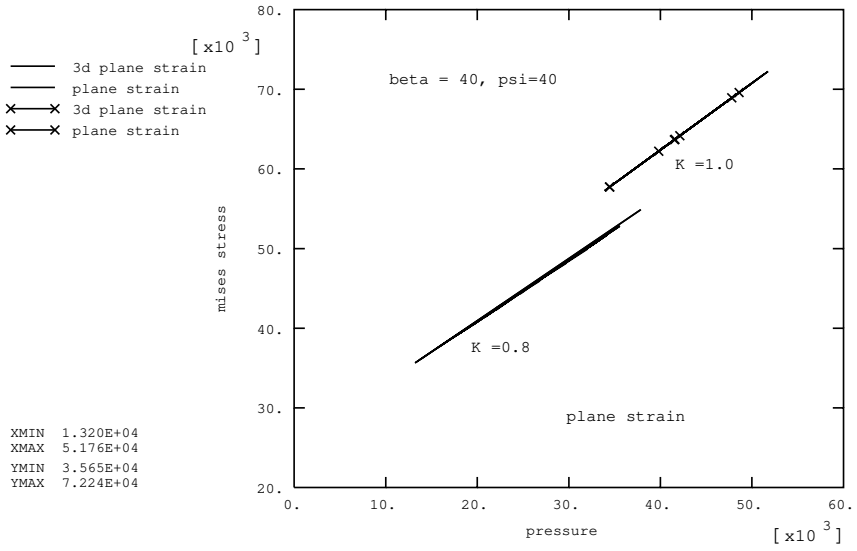


Figure 2: Yield surface in the meridional plane: associated flow, plane strain cases.

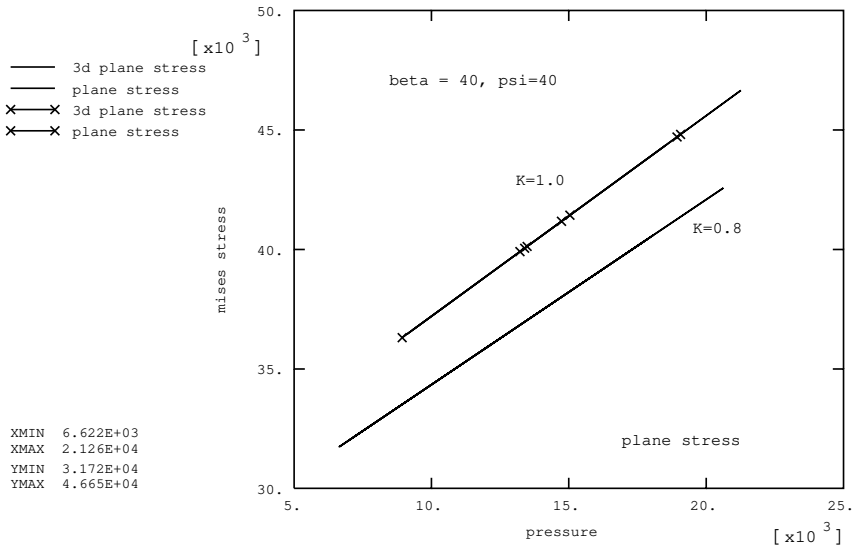


Figure 3: Yield surface in the meridional plane: associated flow, plane stress cases.

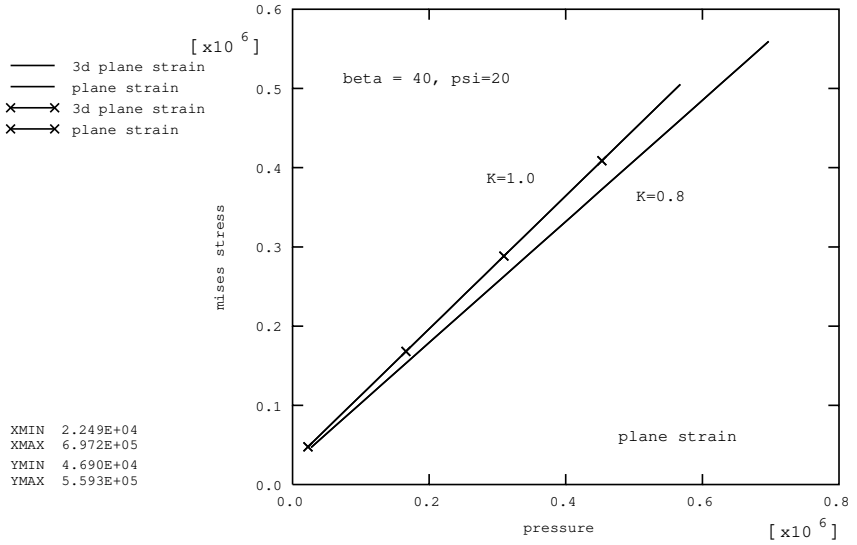


Figure 4: Yield surface in the meridional plane: nonassociated flow, plane strain cases.

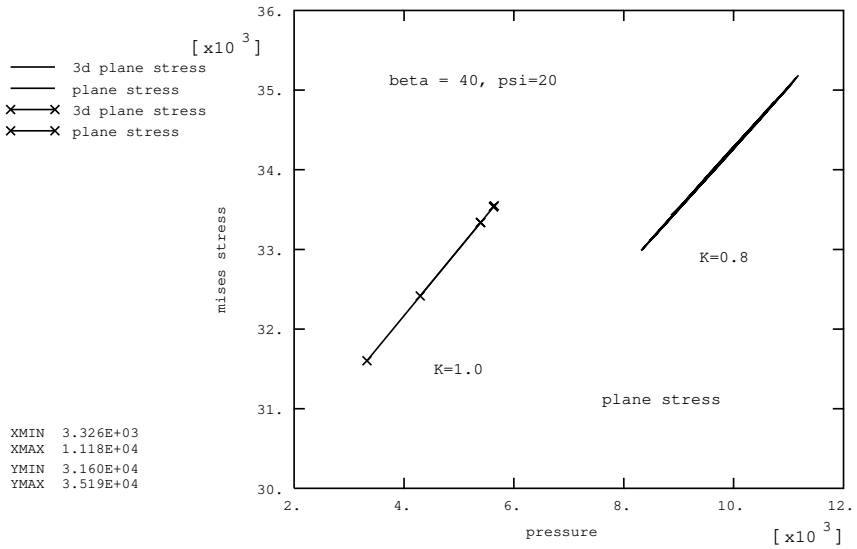


Figure 5: Yield surface in the meridional plane: nonassociated flow, plane stress cases.

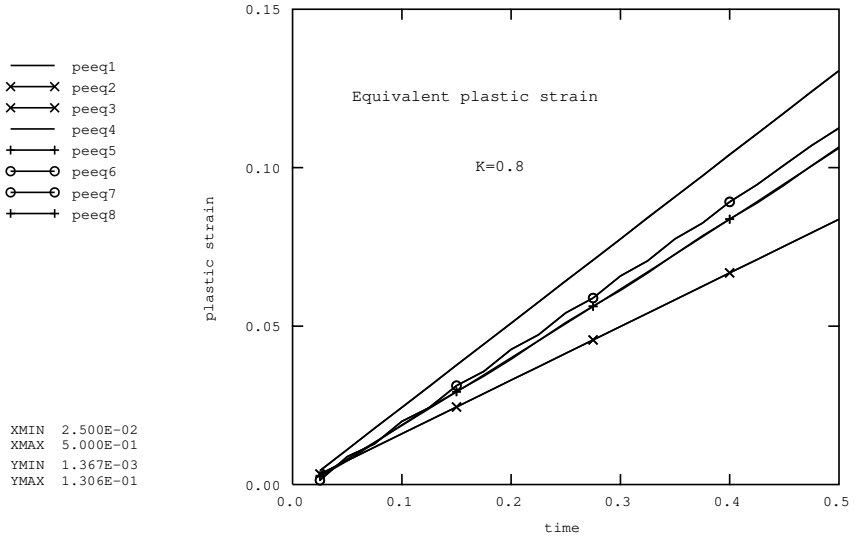


Figure 6: Equivalent plastic strain, $K = 0.8$.

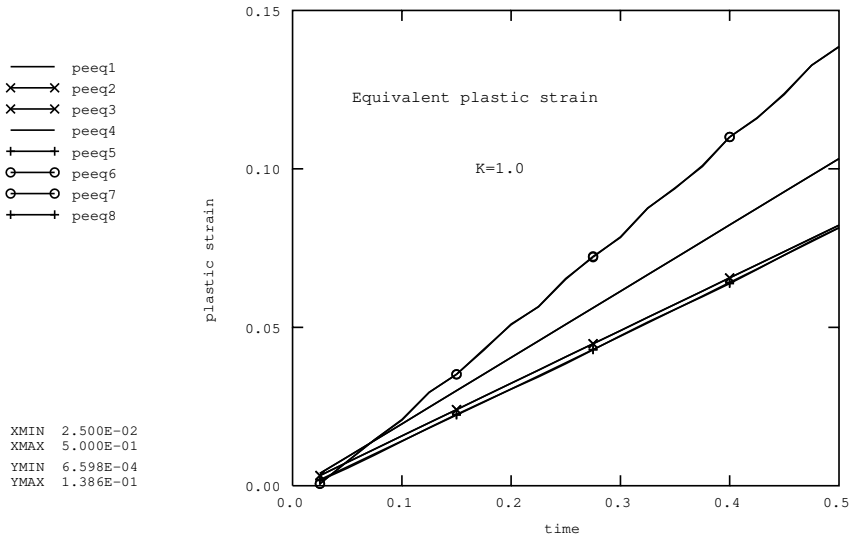


Figure 7: Equivalent plastic strain, $K = 1.0$.

Drucker-Prager/Cap plasticity model

Products: Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D8R CPE4R

Features tested

Drucker-Prager/Cap plasticity model.

Problem description

This problem contains 12 one-element verification problems that are all run in one input file. The problem exercises the Drucker-Prager/Cap plasticity material model. Two different element types are tested (C3D8R, CPE4R). *Figure 1* shows the 12 elements used in the analysis in their original and deformed shapes. The dashed lines represent the original mesh. The 8-node brick element (C3D8R) appears twice in each row: in the second column boundary conditions are applied to constrain the out-of-plane displacement so that the C3D8R element generates plane strain results. No out-of-plane boundary conditions are used for element 1 and element 10 in the first column. For elements 4 and 7 in column one the out-of-plane boundary conditions correspond to hydrostatic tension and compression, respectively. The original length of each side of the elements is $L_0 = 1$.

This example problem is designed to test the following features:

- plane strain and three-dimensional cases
- tension, compression, and simple shear deformations

This is accomplished as described below.

The loading in row (a) represents uniaxial compressive loading in the *x*-direction.

In rows (b) and (c) in *Figure 1* the left and right, top and bottom nodes of each element are given equal and opposite prescribed constant velocities in the *x*- and *y*-directions to generate hydrostatic compressive and tensile loading, respectively.

In row (d) in *Figure 1* the bottom and top nodes of each element are given equal and opposite prescribed constant velocities in the *x*-direction to generate a simple shear loading.

The material's elastic response is assumed to be linear and isotropic, with Young's modulus 30×10^3, Poisson's ratio 0.3, and a density of 0.001. A friction angle of $\beta = 30.0°$ is assumed, and the cap eccentricity parameter is chosen as $R = 0.1$. The transition surface parameter $\alpha = 0.01$ is used.

The initial cap position is taken as $\varepsilon_{\text{vol}}^{pl}(0) = 0.0005$ for rows (b) and (c), and as $\varepsilon_{\text{vol}}^{pl}(0) = 0.002$ for rows (a) and (d). The values of cohesion used for the two cases are $d = 15.0$ and $d = 10.0$, respectively.

Results and discussion

The results obtained from the plane strain elements in all the tests are identical to the corresponding results obtained from the three-dimensional elements where plane strain boundary conditions are applied. The names of the individual curves that appear in the graph legend are a concatenation of the output variable names, an underscore (_), and a number. The number refers to the element number. For example, P-Q_3 refers to the Mises stress versus equivalent pressure stress curve for element 3.

Figure 2 through *Figure 5* show the response of the Drucker-Prager/Cap model. The figures show the two main purposes of the cap surface. Firstly, it bounds the yield surface in hydrostatic compression, thus providing an inelastic hardening mechanism to represent plastic compaction. This behavior is shown in *Figure 3* and *Figure 5* for element 7. The figures show that the pressure stress increases with volume strain according to the cap hardening curve. Once the pressure exceeds the maximum pressure specified on the hardening curve, the response is incompressible. Secondly, the cap surface helps control volume dilatancy by providing softening as a function of the inelastic volume increase created as the material yields on the Drucker-Prager shear failure and transition yield surfaces. This behavior is shown in *Figure 2* and *Figure 5* for element 3. The figures show that during elastic behavior the Mises stress, q, increases at zero pressure stress, p, until first yield. Once the yield surface is reached, inelastic shear deformation occurs, which is accompanied by dilatancy. Since the element is confined (vertical deformation is constrained and plane strain conditions are assumed in the out-of-plane direction), the dilatancy gives rise to an increase in pressure stress. Continuing shearing causes the stress point (p, q) to remain on the yield surface, but to move away from the origin (*Figure 2*). This dilatant behavior also causes the cap surface to move towards the origin (*Figure 5*). Once the stress point meets the cap or transition yield surface, inelastic volume dilatancy ceases and further shearing causes no further increases in Mises or pressure stress.

In hydrostatic tension (element 4) the material loses strength at a pressure stress of $p = d/\tan\beta =$ 26.0 (*Figure 4*). In uniaxial compression (element 10) the stress state, (p, q), satisfies the relation $q = 3p$. Since the material is unconstrained, inelastic volume dilatancy does not give rise to an increase in pressure stress (*Figure 2*), but it causes the cap surface to move towards the origin (*Figure 5*).

This problem tests the Drucker-Prager/Cap plasticity model, but does not provide independent verification of it.

Input files

captests.inp	Input data used in this analysis.

Figures

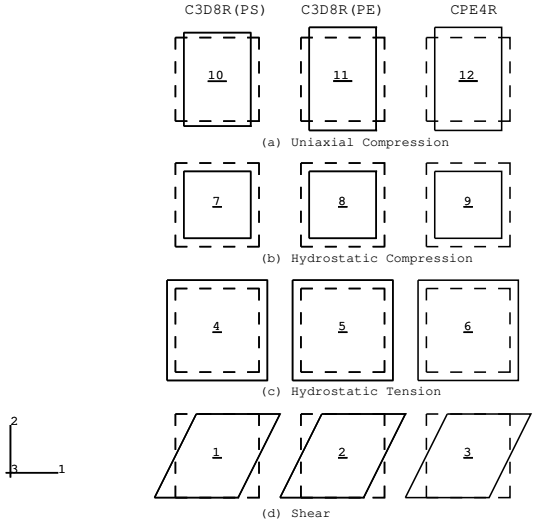


Figure 1: Deformed shape for one element Cap plasticity tests.

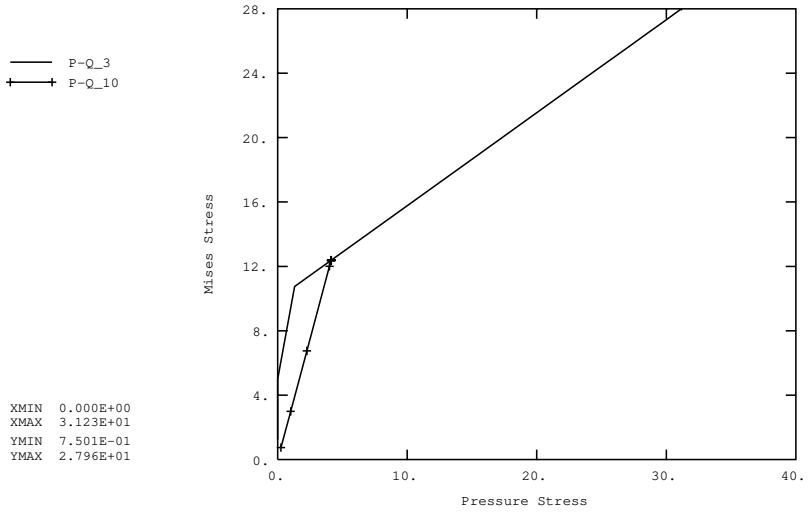


Figure 2: Evolution of stress state in p-q space.

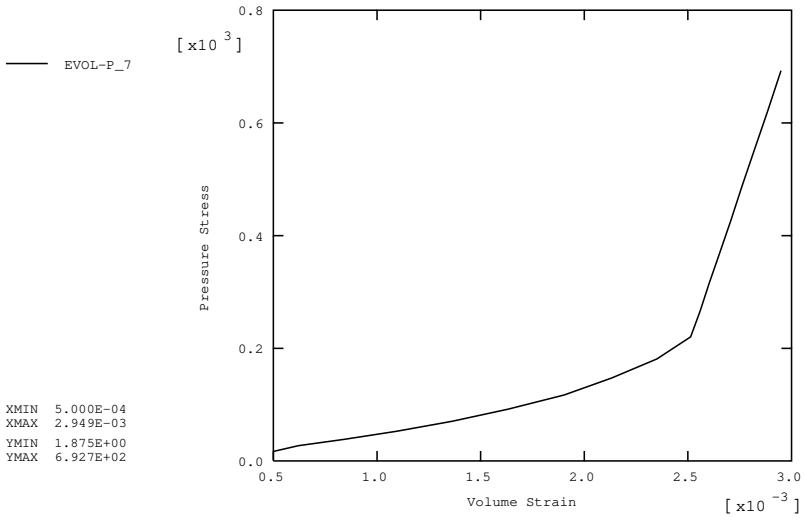


Figure 3: Pressure stress versus volume strain.

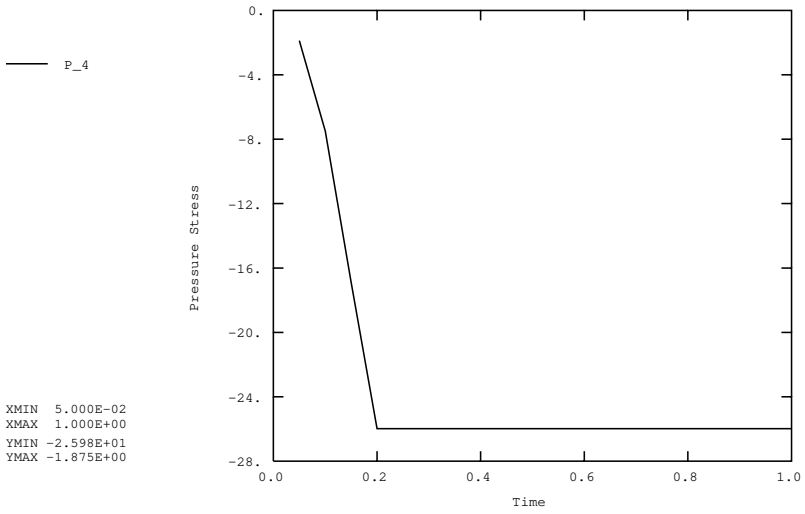


Figure 4: Pressure stress versus time.

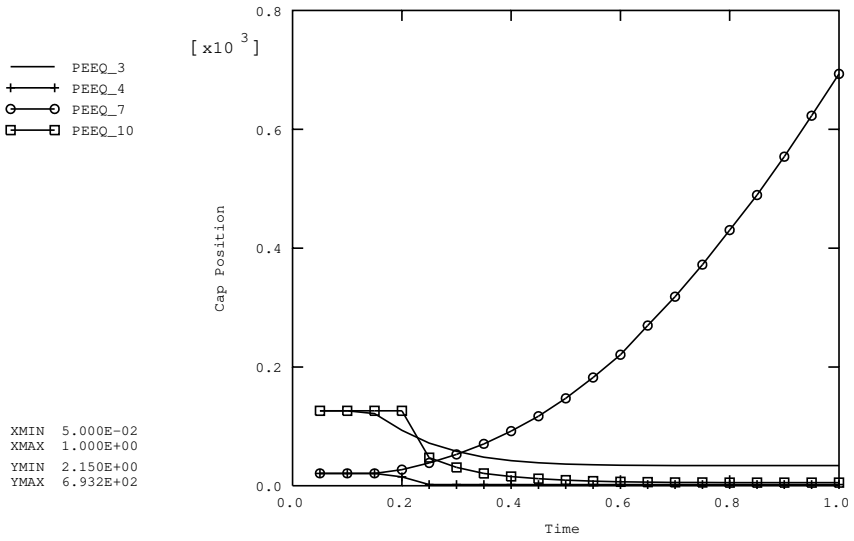


Figure 5: Cap position, p_b, versus time.

Paperboard material model

Products: Abaqus/Standard Abaqus/Explicit

References:

- *Paperboard Material Model*

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D8R S4R

Features tested

Paperboard plasticity model.

Problem description

This verification test contains a set of single-element models under simple loading conditions (uniaxial tension, uniaxial compression, and simple shear). The tests are designed to test the paperboard material model.

Model *Figure 1* shows the elements used to test the in-plane behavior of the paperboard model. *Figure 2* shows the elements used to test the out-of-plane behavior of the paperboard model. A one-element model is used to test the ZD compression behavior.

Material Elastic and plastic properties of the paperboard material used in the tests are listed in *Table 1*, *Table 2*, and *Table 3*. The exponent in the in-plane yield function is set to 4, and the plastic Poisson's ratios in the MD and CD directions are set to 0.5 and 0.133, respectively. The same Poisson's ratios are used for tension and compression.

Results and discussion

The results obtained from the plane stress elements in all the in-plane load tests are the same as those obtained with 3D elements. *Figure 3* shows good agreement between the uniaxial tension results in the MD and CD directions with those in *Xia (2002)*. *Figure 4* shows the stress-strain curve of a ZD compression test under subsequent loading and unloading. The results compare well with those presented in *Andersson (2006)*.

The results that are obtained with Abaqus/Explicit are the same as those obtained with Abaqus/Standard.

Input files

Abaqus/Standard input files

paper_inplane_std.inp	In-plane load tests.
paper_outofplane_std.inp	Out-of-plane load tests.
paper_zdcompress_std.inp	ZD compression test with subsequent loading and unloading.

Abaqus/Explicit input files

paper_inplane_xpl.inp	In-plane load tests.
paper_outofplane_xpl.inp	Out-of-plane load tests.
paper_zdcompress_xpl.inp	ZD compression test with subsequent loading and unloading.

References

- Andersson, T., “A Small Deformation Model for the Elasto-Plastic Behavior of Paper and Paperboard,” *Master's thesis*, 2006.
- Xia, Q. S., “Mechanics of Inelastic Deformation and Delamination in Paperboard,” *PhD thesis, M.I.T*, 2002.

Tables

Table 1: Elastic properties.

E_1 (GPa)	E_2 (GPa)	E_3 (MPa)	υ_{12}	G_{12} (GPa)	G_{13} (MPa)	G_{23} (MPa)
5.6	2.0	100	0.37	4.1	400	400

Table 2: In-plane plastic hardening parameters.

σ_1^0 (MPa)	σ_2^0 (MPa)	σ_3^0 (MPa)	σ_4^0 (MPa)	σ_5^0 (MPa)
12	6.5	6.0	7.3	6.3
A_1 (MPa)	A_2 (MPa)	A_3 (MPa)	A_4 (MPa)	A_5 (MPa)
19	7.4	6.0	6.0	9.0
B_1	B_2	B_3	B_4	B_5
260	160	375	160	310
C_1 (MPa)	C_2 (MPa)	C_3 (MPa)	C_4 (MPa)	C_5 (MPa)
800	160	200	300	225

Table 3: Out-of-plane properties.

μ	r_{orig}	σ_Z^t (MPa)	A_σ (MPa)	B_σ (MPa)	C_σ	A_τ (MPa)	B_τ (MPa)	C_τ
0.026	1.12	0.36	0.4378	-1	6.5	5	3	2

Figures

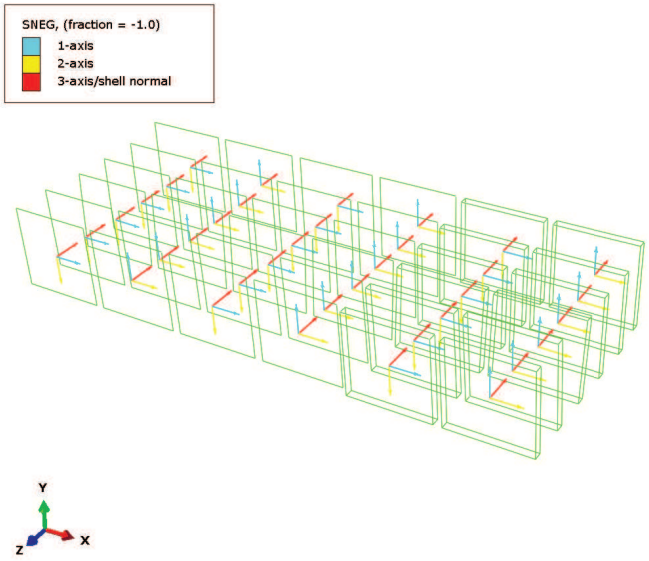


Figure 1: Elements and their material orientations used in the in-plane tests.

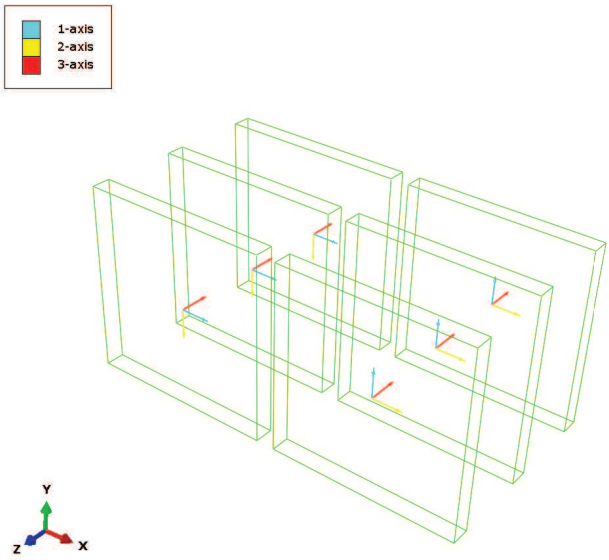


Figure 2: Elements and their material orientations used in the out-of-plane tests.

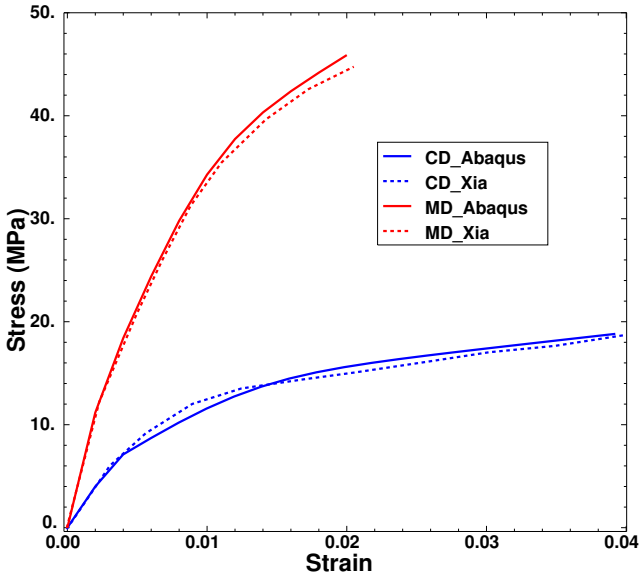


Figure 3: Comparison of stress-strain curves for uniaxial MD and CD tension.

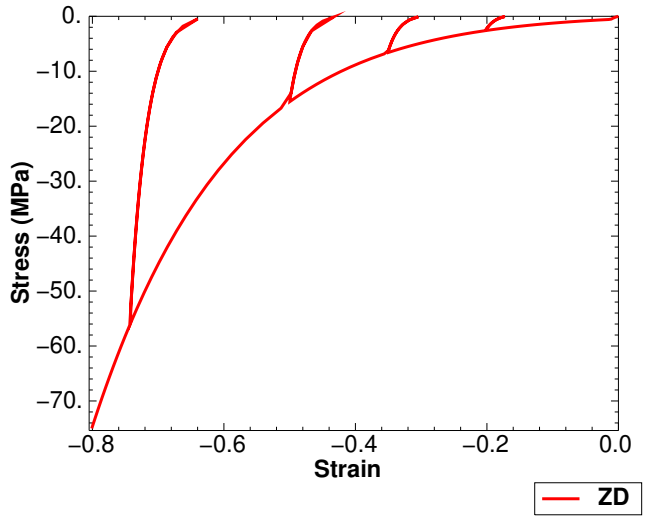


Figure 4: Stress-strain curve for ZD compression in subsequent loading and unloading.

Equation of state material

Products: Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Linear $U_s - U_p$ Hugoniot equation of state

Elements tested

C3D8R CPE4R

Features tested

Linear $U_s - U_p$ equation of state (EOS) material model with plasticity.

Problem description

This verification test consists of a list of single-element models that use either C3D8R or CPE4R elements and are run under simple loading conditions (uniaxial tension, uniaxial compression, and simple shear). The purpose of this example is to test the equation of state material model and its combination with the Mises and Johnson-Cook plasticity models. Two parallel sets of models are studied. The first set uses the linear elastic, linear elastic with Mises plastic, and linear elastic with Johnson-Cook plastic materials. The second set uses the linear $U_s - U_p$ type of EOS, linear $U_s - U_p$ type of EOS with Mises plastic, and linear $U_s - U_p$ type of EOS with Johnson-Cook plastic materials.

For linear elasticity the volumetric response is defined by

$$p = K\varepsilon_{vol},$$

where K is the bulk modulus of the material. The linear $U_s - U_p$ Hugoniot form is

$$p = \frac{\rho_0 c_0^2 \eta}{(1-s\eta)^2}\left(1 - \frac{\Gamma_0 \eta}{2}\right) + \Gamma_0 \rho_0 E_m,$$

where $\eta = 1 - \rho_0/\rho$ is the same as the nominal volumetric strain measure, ε_{vol}. Thus, setting the parameters $s =$ 0.0 and $\Gamma_0 =$ 0.0 gives the simple hydrostatic bulk response, which is identical to the elastic volumetric response. The elastic deviatoric response of the material is defined by the shear modulus.

The elastic material properties are Young's modulus = 207 GPa and Poisson's ratio = 0.29. The initial material density, ρ_0, is 7890 kg/m^3. The equivalent properties for the linear $U_s - U_p$ type of equation of state material model are c_0 = 4563.115 m/s and shear modulus = 80.233 GPa. For models in which plasticity (including both Mises and Johnson-Cook plasticity models) is used, the plastic hardening is chosen to be

$$\overline{\sigma} = 175 + 380\left(\overline{\varepsilon}^{pl}\right)^{0.32},$$

where $\overline{\sigma}$ is the yield stress (in units of MPa) and $\overline{\varepsilon}^{pl}$ is the equivalent plastic strain.

Results and discussion

The results obtained from the analyses that use the EOS material model match the corresponding results obtained from the analyses that use the linear elasticity model. The comparison of the pressure and Mises stresses obtained with the EOS material model (with Johnson-Cook plastic shear response) and the linear elasticity model (with the same Johnson-Cook plastic shear response) using the C3D8R element under uniaxial tension loading are shown in *Figure 1* and *Figure 2*, respectively. The uniaxial compression comparisons are shown in *Figure 3* and *Figure 4*.

Input files

eosshrela.inp	Uniaxial tension test.
eosshrela_pre.inp	Uniaxial compression test.
eosshrela_shr.inp	Simple shear test.
eosshrelainit_shr.inp	Simple shear test with nonzero initial conditions for $\bar{\varepsilon}^{pl}$.

Figures

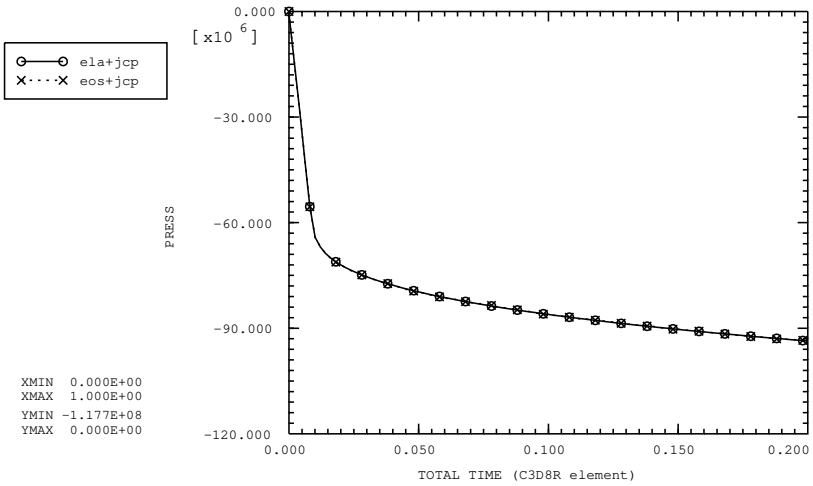


Figure 1: Pressure stress in uniaxial tension: elastic response versus linear $U_s - U_p$ type of equation of state response.

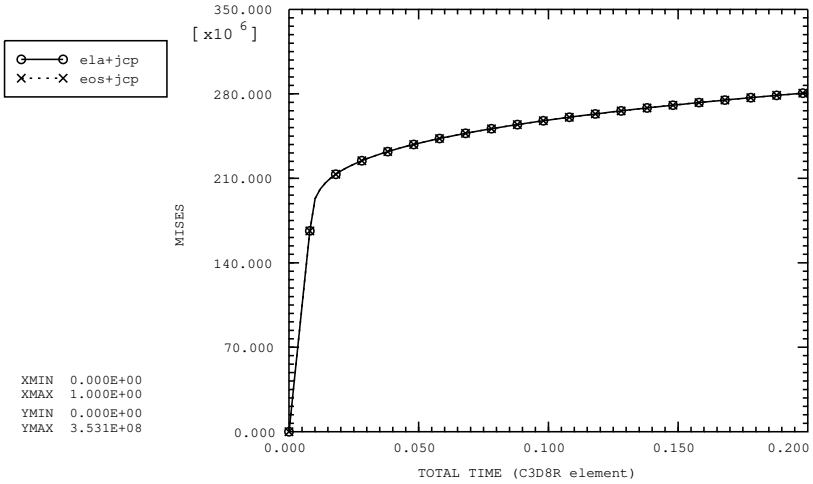


Figure 2: Mises stress in uniaxial tension: elastic response versus linear $U_s - U_p$ type of equation of state response.

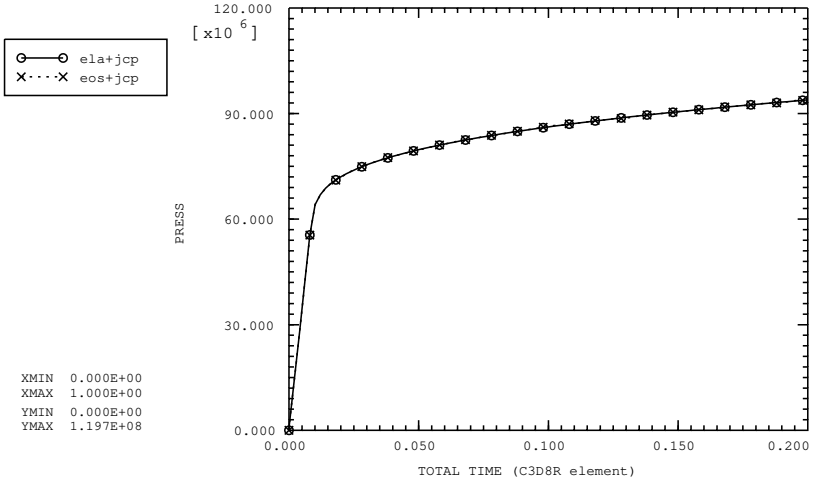


Figure 3: Pressure stress in uniaxial compression: elastic response versus linear $U_s - U_p$ type of equation of state response.

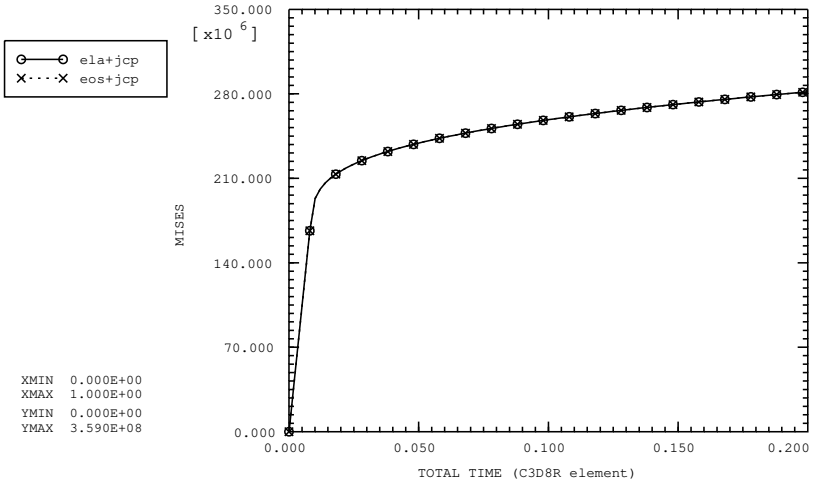


Figure 4: Mises stress in uniaxial compression: elastic response versus linear $U_s - U_p$ type of equation of state response.

Tabulated equation of state

Elements tested

C3D8R CPE4R

Features tested

Tabulated equation of state (EOS) material model with plasticity.

Problem description

This verification test consists of single-element models that use either C3D8R or CPE4R elements and are run under simple loading conditions (uniaxial tension, uniaxial compression, and simple shear). The purpose of this example is to test the tabulated EOS material model and its combination with the Mises and Johnson-Cook plasticity models. Two parallel sets of models are studied. The first set uses the linear elasticity, linear elasticity with Mises plasticity, and linear elasticity with Johnson-Cook plasticity materials. The second set uses the tabulated EOS, tabulated EOS with Mises plasticity, and tabulated EOS with Johnson-Cook plasticity materials.

For linear elasticity the volumetric response is defined by

$$p = -K\varepsilon_{vol},$$

where K is the bulk modulus of the material. The tabulated EOS is linear in energy and assumes the form

$$p = f_1\left(\varepsilon_{\text{vol}}\right) + \rho_0 f_2\left(\varepsilon_{\text{vol}}\right) E_m,$$

where $f_1\left(\varepsilon_{\text{vol}}\right)$ and $f_2\left(\varepsilon_{\text{vol}}\right)$ are functions of the logarithmic volumetric strain ε_{vol} only, with $\varepsilon_{\text{vol}} = \ln(\rho_0/\rho)$, and ρ_0 is the reference density. Thus, setting the functions $f_1\left(\varepsilon_{\text{vol}}\right) = -K\varepsilon_{vol}$ and $f_2\left(\varepsilon_{\text{vol}}\right) =$ 0.0 gives the simple hydrostatic bulk response, which is identical to the elastic volumetric response. The elastic deviatoric response of the material is defined by the shear modulus.

The elastic material properties are Young's modulus = 207 GPa and Poisson's ratio = 0.29. The initial material density, ρ_0, is 7890 kg/m^3. The properties for the tabular EOS material model are computed using K = 164.286 GPa and shear modulus = 80.233 GPa. For models in which plasticity (including both Mises and Johnson-Cook plasticity models) is used, the plastic hardening is chosen to be

$$\overline{\sigma} = 175 + 380\left(\overline{\varepsilon}^{pl}\right)^{0.32},$$

where $\overline{\sigma}$ is the yield stress (in units of MPa) and $\overline{\varepsilon}^{pl}$ is the equivalent plastic strain.

Results and discussion

The results obtained from the analyses that use the EOS material model match the corresponding results obtained from the analyses that use the linear elasticity model.

Input files

eostabshrela.inp	Uniaxial tension test.
eostabshrela_pre.inp	Uniaxial compression test.
eostabshrela_shr.inp	Simple shear test.
eostabshrelainit_shr.inp	Simple shear test with nonzero initial conditions for $\overline{\varepsilon}^{pl}$.

P– equation of state

Elements tested

C3D8R CPE4R

Features tested

$P-\alpha$ equation of state (EOS) material model.

Problem description

This verification test consists of single-element models that use either C3D8R or CPE4R elements and are run under simple loading conditions (uniaxial, hydrostatic, and simple shear). The purpose of this example is to test the $P-\alpha$ equation of state material model and its combination with different models for the deviatoric behavior: linear elastic, Newtonian viscous shear, and Mises and Johnson-Cook plasticity; as well as its combination with different models for the hydrodynamic response of the solid phase: Mie-Grüneisen and tabulated equations of state.

The material properties used for the tests are representative of partially saturated sand. They are summarized below:

Material:

Solid phase

The solid phase is described by a Mie-Grüneisen equation of state:

ρ_{s0}	2070 kg/m^3
C_s	1480 m/sec
s	1.93
Γ_0	0.880

For models using the tabulated equation of state, the functions $f_1\left(\varepsilon_{\text{vol}}\right)$ and $f_2\left(\varepsilon_{\text{vol}}\right)$ are defined such as to provide similar hydrodynamic behavior as the above Mie-Grüneisen equation of estate.

Compaction properties

C_e	600 m/sec
n_0 (α_0)	0.049758 (1.052364)
p_e	0.0 MPa
p_s	6.5 MPa

Viscous shear behavior

μ	5.0E+4

Elastic shear behavior

E	124 MPa
ν	0.3

Plasticity

For models with plastic shear behavior (either Mises or Johnson-Cook plasticity), the plastic hardening is chosen to be

$$\overline{\sigma} = 1.97 + 4.28\left(\overline{\varepsilon}^{pl}\right)^{0.32},$$

where $\overline{\sigma}$ is the yield stress (in units of MPa) and $\overline{\varepsilon}^{pl}$ is the equivalent plastic strain. The plasticity models are used in combination with linear elastic shear behavior.

Results and discussion

The results obtained from the analyses agree well with exact analytical or approximate solutions. The evolution of the distension α with hydrostatic pressure during a cyclic volumetric test is shown in *Figure 5*.

Input files

eospalpha_uni.inp	Uniaxial test.
eospalpha_vol.inp	Cyclic hydrostatic test.
eospalpha_shr.inp	Simple shear test.
eospalphainit_shr.inp	Simple shear test with nonzero initial conditions for $\overline{\varepsilon}^{pl}$.

Figures

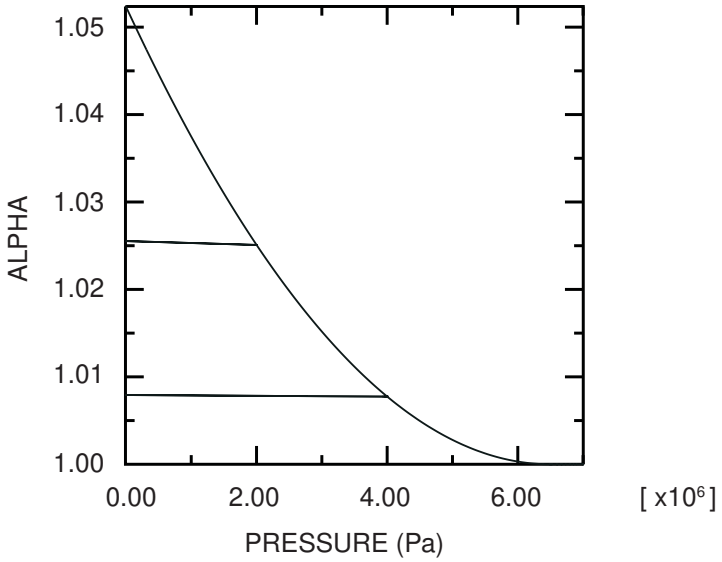


Figure 5: $P - \alpha$ elastic and plastic curves during the cyclic volumetric test.

Viscous shear behavior

Elements tested

C3D8R CPE4R

Features tested

Viscosity models for equation of state materials with viscous shear behavior.

Problem description

This verification test consists of single-element models that use either C3D8R or CPE4R elements and are run under simple shear loading conditions. The purpose of this example is to test the different viscosity models for both Newtonian and non-Newtonian fluids. The hydrodynamic response of the material is described by the Mie-Grüneisen equation of state in all cases. Some tests include thermorheologically simple temperature-dependent viscosity using the Arrhenius form.

The material properties used for the tests are summarized below:

Material:

Hydrodynamic properties

The hydrodynamic response described by a Mie-Grüneisen equation of state:

ρ_{s0}	2070 kg/m^3
C_s	1480 m/sec
s	1.93

Γ_0	0.880

Viscous properties

The properties for each of the tested viscosity models are given below:

Mat1:

Newtonian viscosity:

η	1 Pa sec

Mat2:

Power Law viscosity:

k	2.173 Pa $(sec)^n$
n	0.392
η_{max}	1 Pa sec
η_{min}	0.1 Pa sec

Mat3:

Carreau-Yasuda viscosity:

η_0	1 Pa sec
η_∞	0.1 Pa sec
λ	0.11 sec
n	0.392
a	0.644

Mat4:

Cross viscosity:

η_0	1 Pa sec
η_∞	0.1 Pa sec
λ	0.11 sec
n	0.392

Mat5:

Herschel-Bulkley viscosity:

η_0	1 Pa sec
τ_0	3.59 Pa
k	2.173 Pa (sec)^n
n	0.392

Mat6:

Ellis-Meter viscosity:

η_0	1 Pa sec
η_∞	0.1 Pa sec
$\tau_{1/2}$	5.665 Pa
n	0.392

Mat7:

Powell-Eyring viscosity:

η_0	1 Pa sec
η_∞	0.1 Pa sec
λ	0.11 sec

Mat8:

Tabular viscosity:

η (Pa sec)	$\dot{\gamma}$ (sec^{-1})
1.00000	0.0
0.83383	1.0
0.76532	2.0
0.71776	3.0
0.68112	4.0
0.65134	5.0
0.62631	6.0
0.60477	7.0
0.58593	8.0
0.56921	9.0
0.55422	10.0
0.54066	11.0
0.52830	12.0
0.51697	13.0

0.50652	14.0
0.49684	15.0

Mat9:

User-defined Cross viscosity. The viscosity is expressed as

$$\eta = \frac{\eta_0}{1+(\lambda\dot{\gamma})^{1-n}},$$

η_0	1 Pa sec
λ	0.11 sec
n	0.392

TRS properties

Arrhenius form:

E_0	109100 joule/mole
θ_0	308 kelvin
θ^Z	0 kelvin
R	8.31434 joule/(mole kelvin)

Results and discussion

The results obtained from the analyses agree well with exact analytical or approximate solutions.

Input files

eosshrvisc.inp	Simple shear test.
eosshrvisctrs.inp	Material with Arrhenius TRS properties. Simple shear test.
eosshrvisc.f	User subroutine *VUVISCOSITY* for the user-defined Cross viscosity model used in eosshrvisc.inp and eosshrvisctrs.inp.

Nonlinear viscoelastic shear behavior

Elements tested

C3D8R CPE4R

Features tested

Nonlinear viscoelastic models for equation of state materials using a combination of elastic and viscous shear behavior.

Problem description

This verification test consists of single-element models that use either C3D8R or CPE4R elements and are run under simple shear loading conditions. The purpose of this example is to test the family of nonlinear viscoelastic shear models defined using a combination of elastic and viscous shear behaviors with different Newtonian and non-Newtonian shear viscosities. The hydrodynamic response of the material is described by the Mie-Grüneisen equation of state in all cases. Some tests include thermorheologically simple temperature-dependent viscosity using the Arrhenius form.

The material properties used for the tests are the same as in *Viscous shear behavior* except that an elastic shear modulus is defined: μ = 1000Pa.

Results and discussion

The results obtained from the analyses agree well with exact analytical or approximate solutions.

Input files

eosshrelasvisc.inp	Simple shear test.
eosshrelasvisctrs.inp	Material with Arrhenius TRS properties. Simple shear test.
eosshrvisc.f	User subroutine *VUVISCOSITY* for the user-defined Cross viscosity model used in `eosshrelasvisc.inp` and `eosshrelasvisctrs.inp`.

Pressure-dependent shear plasticity

Elements tested

C3D8R CPE4R CAX4R

Features tested

Equation of state (EOS) material model with pressure-dependent (Drucker-Prager) shear plasticity.

Problem description

This verification test consists of single-element models that use either C3D8R, CPE4R, or CAX4R elements and are run under simple loading conditions (uniaxial tension, uniaxial compression, and simple shear). The purpose of this example is to test the combination of EOS models for the volumetric response of the material with the extended Drucker-Prager pressure-dependent plasticity models for the shear response. Some of the models also include Johnson-Cook strain-rate dependence in the plasticity definition.

Results and discussion

The results agree well with exact analytical or approximate solutions.

Input files

eosjcratedpexpuni3d.inp	Uniaxial tension test, Johnson-Cook strain-rate dependence, Drucker-Prager plasticity with exponent form shear criterion, C3D8R element.
eosjcratedpexpunicpe.inp	Uniaxial tension test, Johnson-Cook strain-rate dependence, Drucker-Prager plasticity with exponent form shear criterion, CPE4R element.
eosjcratedpexpuniaxi.inp	Uniaxial tension test, Johnson-Cook strain-rate dependence, Drucker-Prager plasticity with exponent form shear criterion, CAX4R element.

eosjcratedphypuni3d.inp	Uniaxial tension test, Johnson-Cook strain-rate dependence, Drucker-Prager plasticity with hyperbolic shear criterion, C3D8R element.
eosjcratedphypunicpe.inp	Uniaxial tension test, Johnson-Cook strain-rate dependence, Drucker-Prager plasticity with hyperbolic shear criterion, CPE4R element.
eosjcratedphypuniaxi.inp	Uniaxial tension test, Johnson-Cook strain-rate dependence, Drucker-Prager plasticity with hyperbolic shear criterion, CAX4R element.
eosdruckerprager.inp	Uniaxial tension test, C3D8R and CPE4R elements.
eosdruckerprager_pre.inp	Uniaxial compression test, C3D8R and CPE4R elements.
eosdruckerprager_shr.inp	Simple shear test, C3D8R and CPE4R elements.

User-defined equation of state

Elements tested

C3D8R CPE4R

Features tested

User-defined equation of state (EOS) material model with plasticity.

Problem description

This verification test consists of single-element models that use either C3D8R or CPE4R elements and are run under simple loading conditions (uniaxial tension, uniaxial compression, and simple shear). The purpose of this example is to test the user-defined EOS material model (user subroutine *VUEOS*) and its combination with the Mises and Johnson-Cook plasticity models. Two parallel sets of models are studied. The first set uses the linear elasticity, linear elasticity with Mises plasticity, and linear elasticity with Johnson-Cook plasticity materials. The second set uses the user-defined EOS, user-defined EOS with Mises plasticity, and user-defined EOS with Johnson-Cook plasticity materials.

For linear elasticity the volumetric response is defined by

$$p = -K\varepsilon_{vol},$$

where K is the bulk modulus of the material. To obtain the same elastic volumetric response with the user-defined EOS, the pressure update inside user subroutine *VUEOS* is

$$p = -K\varepsilon_{vol} = -K\ln(\rho_0/\rho),$$

where ρ_0 is the reference density. The user subroutine needs to return the derivative of pressure with respect to density, $\partial p/\partial \rho$, which is needed for the evaluation of the effective moduli of the material that enters the stable time calculation. User subroutine *VUEOS* also returns the derivative of pressure with respect to the energy, $\partial p/\partial E_m$, which is usually needed to solve the nonlinear pressure-energy dependency using the Newton method. In the case considered here, these quantities are

$$\partial p/\partial \rho = K/\rho, \partial p/\partial E_m = 0.$$

The elastic deviatoric response of an equation of state material can be defined by using the **ELASTIC*, TYPE=SHEAR option.

The elastic material properties are Young's modulus = 207 GPa and Poisson's ratio = 0.29. The initial material density, ρ_0, is 7890 kg/m^3. The properties for the tabular EOS material model are computed using K = 164.286 GPa and shear modulus = 80.233 GPa. For models in which plasticity is used (including both Mises and Johnson-Cook plasticity models), the plastic hardening is chosen to be

$$\overline{\sigma} = 175 + 380\left(\overline{\varepsilon}^{pl}\right)^{0.32},$$

where $\overline{\sigma}$ is the yield stress (in units of MPa) and $\overline{\varepsilon}^{pl}$ is the equivalent plastic strain.

Results and discussion

The results obtained from the analyses that use the EOS material model match the corresponding results obtained from the analyses that use the linear elasticity model.

Input files

eosusershrela.inp	Uniaxial tension test.
eosusershrela_pre.inp	Uniaxial compression test.
eosusershrela_shr.inp	Simple shear test.
eosusershrelainit_shr.inp	Simple shear test with nonzero initial conditions for $\overline{\varepsilon}^{pl}$.
vueos_tab.f	User subroutine *VUEOS* used in the input files listed above.

Progressive damage and failure of ductile metals

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Ductile, Johnson-Cook, Hosford-Coulomb, and shear criteria

Elements tested

T2D2 T3D2 B21 B31 SAX1 C3D8 C3D8R SC8R S4 S4R S4RS CPS4R CPE4R CAX4R M3D4R M3D4

Features tested

Ductile, shear, and Hosford-Coulomb damage initiation criteria are tested for the following material models: Mises plasticity; Hill plasticity; Drucker-Prager plasticity; and, in Abaqus/Explicit, equation of state with Johnson-Cook plasticity. The Johnson-Cook criterion, a special case of the ductile criterion, is also tested with the following material models: Mises plasticity, Hill plasticity, Johnson-Cook plasticity, Drucker-Prager plasticity, and equation of state with Mises plasticity. For the ductile and the shear damage initiation criteria, the capability to specify initial conditions on the damage initiation measures is tested. The elastic response of the material is defined using either the linear elastic material behavior or the hyperelastic behavior.

Problem description

This verification test consists of a set of single-element models subjected to biaxial tension; an exception is the truss and beam elements, which are loaded by uniaxial tension. For each material model only those element types supported for that model are used. The ductile criterion is specified in terms of the plastic strain at the onset of damage as a tabular function of the stress triaxiality and the equivalent plastic strain rate. In Abaqus/Explicit the ductile criterion can also be defined as a tabular function of Lode angle. The Johnson-Cook criterion (available only in Abaqus/Explicit) is specified in terms of failure parameters d_1–d_5, the reference strain rate $\dot{\varepsilon}_0$, the melting temperature, and the transition temperature. The shear criterion is specified in terms of the plastic strain at the onset of damage as a tabular function of the shear stress ratio and the equivalent plastic strain rate. The Hosford-Coulomb criterion (available only in Abaqus/Standard) is specified in terms of failure parameters a, b, c, and n. The damage evolution law (available only in Abaqus/Explicit) is specified in terms of the equivalent plastic displacement or in terms of the fracture energy dissipation. A maximum degradation of 0.75 is set. The default failure choice (i.e., element deletion) is used in all tests in this subsection.

The tests for initial conditions on the damage initiation measures do not subject the elements to any deformation. Instead, these tests verify only that the specified initial conditions are output without any modifications.

Results and discussion

The damage initiation criteria computed by Abaqus agree with the values obtained analytically. For models that include damage evolution, the material degradation starts when the equivalent plastic strain reaches the specified damage initiation criterion. The damage variable evolves according to the evolution law specified in terms of the plastic displacement or energy dissipation. The element is deleted from the mesh once all the integration points at any one section of an element fail; the element output variable STATUS will then be set to zero.

Input files

Ductile criterion

damage_ductile_mises.inp	Ductile criterion, Mises plasticity.
damage_ductile_mises_ic.inp	Ductile criterion, test initial conditions.
damage_ductile_lode_mises.inp	Ductile criterion with Lode angle dependency, Mises plasticity.
damage_ductile_hill.inp	Ductile criterion, Hill plasticity.
damage_ductile_dp.inp	Ductile criterion, Drucker-Prager plasticity.
damage_ductile_eos.inp	Ductile criterion, equation of state with Johnson-Cook plasticity.
damage_ductile_mises_std.inp	Ductile criterion, Mises plasticity in Abaqus/Standard.
damage_ductile_fefp_std.inp	Ductile criterion, Hencky hyperelastic material, Mises plasticity in Abaqus/Standard.
damage_ductile_mises_ic_std.inp	Ductile criterion in Abaqus/Standard, test initial conditions.
damage_ductile_fefp_ic_std.inp	Ductile criterion, Hencky hyperelastic model, Mises plasticity in Abaqus/Standard, test initial conditions.
johnsoncook_dmg_s.inp	Ductile criterion, Johnson-Cook plasticity in Abaqus/Standard.
damage_ductile_hill_std.inp	Ductile criterion, Hill plasticity in Abaqus/Standard.
damage_ductile_dp_std.inp	Ductile criterion, Drucker-Prager plasticity in Abaqus/Standard.

Johnson-Cook criterion

damage_jc_mises.inp	Johnson-Cook criterion, Mises plasticity.
damage_jc_hill.inp	Johnson-Cook criterion, Hill plasticity.
damage_jc_jc.inp	Johnson-Cook criterion, Johnson-Cook plasticity.
damage_jc_dp.inp	Johnson-Cook criterion, Drucker-Prager plasticity.
damage_jc_eos.inp	Johnson-Cook criterion, equation of state with Mises plasticity.

Hosford-Coulomb criterion

damage_ductile_mises_hc_std.inp	Hosford-Coulomb criterion, Mises plasticity in Abaqus/Standard.
damage_ductile_mises_ic_hc_std.inp	Hosford-Coulomb criterion in Abaqus/Standard, test initial conditions.
johnsoncook_dmg_hc_s.inp	Hosford-Coulomb criterion, Johnson-Cook plasticity in Abaqus/Standard.
damage_ductile_hill_hc_std.inp	Hosford-Coulomb criterion, Hill plasticity in Abaqus/Standard.
damage_ductile_dp_hc_std.inp	Hosford-Coulomb criterion, Drucker-Prager plasticity in Abaqus/Standard.

Shear criterion

damage_shear_mises.inp	Shear criterion, Mises plasticity.
damage_shear_mises_ic.inp	Shear criterion, test initial conditions.
damage_shear_hill.inp	Shear criterion, Hill plasticity.
damage_shear_dp.inp	Shear criterion, Drucker-Prager plasticity.
damage_shear_eos.inp	Shear criterion, equation of state with Johnson-Cook plasticity.
damage_shear_mises_std.inp	Shear criterion, Mises plasticity in Abaqus/Standard.

damage_shear_mises_ic_std.inp	Shear criterion in Abaqus/Standard, test initial conditions.
damage_shear_fefp_ic_std.inp	Shear criterion, Hencky hyperelastic material, Mises plasticity in Abaqus/Standard, test initial conditions.
damage_shear_hill_std.inp	Shear criterion, Hill plasticity in Abaqus/Standard.
damage_shear_dp_std.inp	Shear criterion, Drucker-Prager plasticity in Abaqus/Standard.

Forming limit diagram (FLD) criterion and forming limit stress diagram (FLSD) criterion

Elements tested

SC8R S4 S4R S4RS CPS4R M3D4 M3D4R

Features tested

The FLD and FLSD damage initiation criteria are tested on elements with a plane stress formulation for the following material models: Mises plasticity; Hill plasticity; Drucker-Prager plasticity; and, in Abaqus/Explicit, for Johnson-Cook plasticity.

Problem description

This verification test consists of a set of single-element models subjected to equibiaxial tension. The FLD criterion is specified in terms of the maximum in-plane principal strain at damage initiation as a tabular function of the minimum in-plane principal strain. The FLSD criterion is specified in terms of the maximum in-plane principal limit stress as a tabular function of the minimum in-plane principal stress. In Abaqus/Explicit input files the damage evolution law is specified in terms of the equivalent plastic displacement or in terms of the fracture energy dissipation. A maximum degradation of 0.75 is used. The default failure choice (i.e., element deletion) is used in all tests in this subsection.

Results and discussion

For the FLD criterion material degradation starts when the maximum in-plane principal strain reaches the major limit strain according to the specified forming limit curve. For the FLSD criterion material degradation starts when the maximum in-plane principal stress reaches the major limit stress according to the specified forming limit stress curve. The damage variable evolves according to the evolution law specified in terms of the plastic displacement or energy dissipation.

Input files

FLD criterion

damage_fld_mises.inp	FLD criterion, Mises plasticity.
damage_fld_hill.inp	FLD criterion, Hill plasticity.
damage_fld_dp.inp	FLD criterion, Drucker-Prager plasticity.
damage_fld_jc.inp	FLD criterion, Johnson-Cook plasticity.
damage_fld_mises_std.inp	FLD criterion, Mises plasticity in Abaqus/Standard.
damage_fld_hill_std.inp	FLD criterion, Hill plasticity in Abaqus/Standard.
damage_fld_dp_std.inp	FLD criterion, Drucker-Prager plasticity in Abaqus/Standard.

FLSD criterion

damage_flsd_mises.inp	FLSD criterion, Mises plasticity.
damage_flsd_hill.inp	FLSD criterion, Hill plasticity.

damage_flsd_dp.inp	FLSD criterion, Drucker-Prager plasticity.
damage_flsd_jc.inp	FLSD criterion, Johnson-Cook plasticity.
damage_flsd_mises_std.inp	FLSD criterion, Mises plasticity in Abaqus/Standard.
damage_flsd_hill_std.inp	FLSD criterion, Hill plasticity in Abaqus/Standard.
damage_flsd_dp_std.inp	FLSD criterion, Drucker-Prager plasticityin Abaqus/Standard.

Marciniak-Kuczynski (M-K) criterion

Elements tested

SC8R S4 S4R S4RS CPS4R M3D4R M3D4

Features tested

The M-K damage initiation criterion is tested for Mises plasticity in Abaqus/Explicit.

Problem description

First, a set of single elements with plane stress formulation is loaded under equibiaxial tension to test the M-K damage initiation criterion for different element types. The material properties for this test correspond to a steel alloy modeled with rate-dependent Mises plasticity. The initial imperfection size is defined as a tabular function of the angular direction. The M-K criterion is specified in terms of the limit ratio of the deformation in the groove (thickness imperfection) relative to the nominal deformation outside the groove.

In addition, to demonstrate the capability of the M-K analysis in predicting forming limit diagrams for an aluminum alloy, a set of parametric studies are performed to evaluate the effect of strain paths on the FLDs using S4R elements. In these studies an aluminum alloy (AA 5754–O) is modeled using isotropic Mises plasticity with

Nadai hardening: $\sigma_y = C\left(\varepsilon^{pl} + \varepsilon_0\right)^n$, with $C = 446.3$MPa, $n = 0.292$, and $\varepsilon_0 = 0.0039$. The initial imperfection size is assumed to be 0.9999 in these studies. The number of virtual imperfections is set to 100. A set of analyses are performed with the ratio between the major and minor principal strain parameterized and kept constant throughout each individual analysis, which generates the FLD curve without prestrain. To evaluate the effect of the loading paths on the FLDs, two more sets of studies are performed in which the material is initially prestrained (either with plane strain or equibiaxial loading) and subsequently subjected to the same type of proportional loading as in the case without prestrain.

Results and discussion

Material degradation starts when the ratio of the deformation in the groove relative to the nominal deformation reaches the specified critical value. The damage variable evolves according to the evolution rule specified in terms of the plastic displacement or energy dissipation.

Figure 1 shows the FLD curves predicted with the M-K analyses for the three sets of parametric studies described above, along with a typical loading path involved in each study. The predicted FLD curve with no prestrain matches the analytical criterion suggested by *Hill (1952)* in the left side of the FLD curve (drawing region). The 10% plane strain prestrain shifts the FLD curve upward and, thus, increases the forming limit in both the drawing region and the stretching region. The 10% equibiaxial prestrain moves the FLD curve downward and to the right; therefore, the forming limit is increased in some regions while lowered in others. These results suggest that the FLDs strongly depend on the loading paths prior to reaching the localization point.

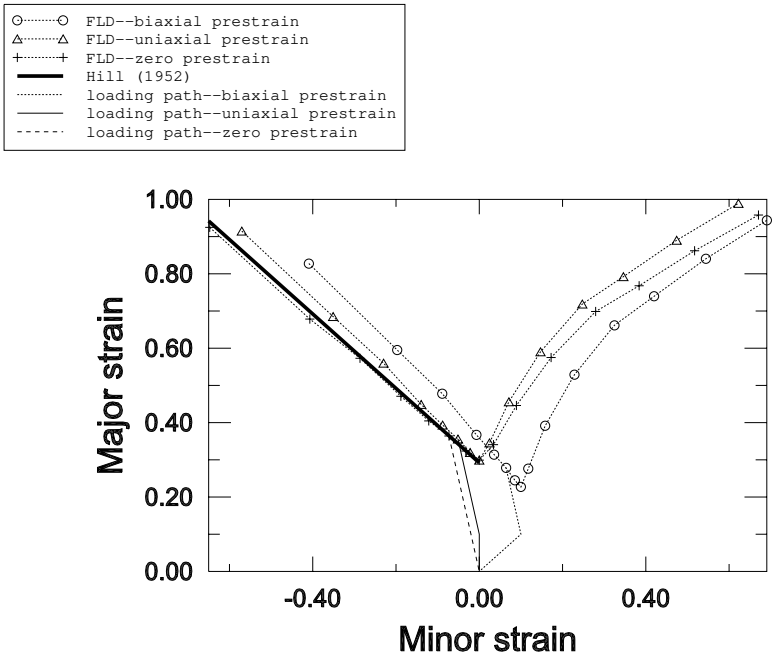


Figure 1: Forming limit diagram.

Input files

damage_mk_mises.inp	M-K criterion; steel alloy; rate-dependent Mises plasticity; SC8R, S4, S4R, S4RS, CPS4R, M3D4R, and M3D4 elements.

Prediction of FLDs using S4R elements

damage_prestrain_no.inp	Template file for parametric study of aluminum alloy with zero prestrain.
damage_prestrain_no.psf	Script file for parametric study of aluminum alloy with zero prestrain.
damage_prestrain_uniaxial.inp	Template file for parametric study of aluminum alloy with uniaxial prestrain.
damage_prestrain_uniaxial.psf	Script file for parametric study of aluminum alloy with uniaxial prestrain.
damage_prestrain_biaxial.inp	Template file for parametric study of aluminum alloy with biaxial prestrain.
damage_prestrain_biaxial.psf	Script file for parametric study of aluminum alloy with biaxial prestrain.

Müschenborn-Sonne forming limit diagram (MSFLD)

Elements tested

SC8R S4R S4RS CPS4R M3D4R M3D4

Features tested

The MSFLD damage initiation criterion is tested for Mises plasticity. The capability to specify initial conditions on the damage initiation measure is also tested.

Problem description

A set of single elements with a plane stress formulation is loaded under equibiaxial tension to test the MSFLD damage initiation criterion for different element types. The MSFLD criterion is specified in terms of the maximum in-plane principal strain at damage initiation as a tabular function of the minimum in-plane principal strain (FLD definition) or in terms of the equivalent plastic strain at damage initiation as a tabular function of the ratio of principal strain rates (MSFLD definition).

To demonstrate the capability of the MSFLD criterion in predicting failure for nonlinear strain paths, a number of numerical simulations of two-step forming processes have been carried out in Abaqus/Explicit using the MSFLD criterion as well as the M-K criterion. Each of the two forming steps follows a linear path with constant principal strain rate ratio α, but there can be a jump in the value of α from the first step to second step; therefore, the overall deformation path is not linear. Based on the value of α throughout the first step and the value of equivalent plastic strain at the end of the first step, these simulations are grouped into five sets: within each set, individual simulations differ only in the value of α during the second step. The same material model described in the last section (AA 5754–O) has also been used here.

The test for initial conditions on the damage initiation measure does not subject the elements to any deformation. Instead, this test verifies only that the specified initial conditions are output without any modifications.

Results and discussion

As shown in *Figure 1*, the forming limit diagrams in the space of major versus minor principal strain (FLD representation) strongly depend on the loading path. However, by representing the same data from the M-K analysis in the space of equivalent plastic strain versus the ratio of principal strain rates (MSFLD representation), those three curves fall onto the same curve as shown in *Figure 2*. This curve has been used to define the MSFLD criterion for the two-step numerical simulations described above. The points of initiation of necking predicted by the M-K criterion for each of the two-step forming processes that are being considered are shown in *Figure 3*, *Figure 4*, *Figure 5*, *Figure 6*, and *Figure 7*. In these figures the solid symbols represent the material state at the end of the first forming step (i.e., the starting point for the second loading step) and the corresponding hollow symbols represent the points of initiation of necking along different loading paths during the second step. The same data are also plotted in *Figure 8* in the $\bar{\varepsilon}^{pl}$-α diagram. The dashed lines in *Figure 8* connect the necking points obtained using the MSFLD criterion for each of the two-step forming processes. As shown in the figure, in most situations the necking predictions based on the MSFLD compare remarkably well with those based on the more expensive M-K analysis. The only case observed in this figure in which the M-K and MSFLD criteria are not in close agreement corresponds to the predeformation of $\alpha = 0.3$ with higher equivalent plastic strain (solid square). In this case the MSFLD criterion slightly over predicts the forming limits for deformation states on the right side of the curve. This situation may be expected to occur when the deformation state of the material gets very close to the forming limit curve sometime during the loading history and is subsequently strained in a direction along which it can sustain further deformation. However, this mismatch can be accounted for through precalibration and the use of a safety factor. These results indicate that the onset of necking instability occurs when a new deformation state in the equivalent plastic strain versus principal strain rate ratio space either lies on the forming limit curve or, upon sudden change in the strain rate ratio, a line connecting the states just before and after the change in strain rate ratio crosses the forming limit diagram. This example demonstrates the capability of the MSFLD criterion in predicting necking even for the nonlinear strain paths.

Input files

damage_msfld_msfld_mises.inp	MSFLD criterion with MSFLD definition, Mises plasticity.
damage_msfld_msfld_mises_ic.inp	MSFLD criterion, test initial conditions.
damage_msfld_fld_mises.inp	MSFLD criterion with FLD definition, Mises plasticity.
damage_msfld_msfld_mises_std.inp	MSFLD criterion with MSFLD definition, Mises plasticity in Abaqus/Standard.

damage_msfld_msfld_mises_ic_std.inp	MSFLD criterion in Abaqus/Standard, test initial conditions.
damage_msfld_fld_mises_std.inp	MSFLD criterion with FLD definition, Mises plasticity in Abaqus/Standard.

Comparison of failure predictions from MSFLD criterion versus those from M-K analysis

damage_msfld_p0p3_lower.inp	Template file for parametric study using MSFLD criterion with starting point of $\alpha = 0.3$ and lower equivalent plastic strain.
damage_msfld_p0p3_lower.psf	Script file for parametric study using MSFLD criterion with starting point of $\alpha = 0.3$ and lower equivalent plastic strain.
damage_msfld_p0p3_higher.inp	Template file for parametric study using MSFLD criterion with starting point of $\alpha = 0.3$ and higher equivalent plastic strain.
damage_msfld_p0p3_higher.psf	Script file for parametric study using MSFLD criterion with starting point of $\alpha = 0.3$ and higher equivalent plastic strain.
damage_msfld_m0p6_lower.inp	Template file for parametric study using MSFLD criterion with starting point of $\alpha = -0.6$ and lower equivalent plastic strain.
damage_msfld_m0p6_lower.psf	Script file for parametric study using MSFLD criterion with starting point of $\alpha = -0.6$ and lower equivalent plastic strain.
damage_msfld_m0p6_higher.inp	Template file for parametric study using MSFLD criterion with starting point of $\alpha = -0.6$ and higher equivalent plastic strain.
damage_msfld_m0p6_higher.psf	Script file for parametric study using MSFLD criterion with starting point of $\alpha = -0.6$ and higher equivalent plastic strain.
damage_msfld_m0p4.inp	Template file for parametric study using MSFLD criterion with starting point of $\alpha = -0.4$.
damage_msfld_m0p4.psf	Script file for parametric study using MSFLD criterion with starting point of $\alpha = -0.4$.
damage_mk_p0p3_lower.inp	Template file for parametric study using M-K analysis with starting point of $\alpha = 0.3$ and lower equivalent plastic strain.
damage_mk_p0p3_lower.psf	Script file for parametric study using M-K analysis with starting point of $\alpha = 0.3$ and lower equivalent plastic strain.
damage_mk_p0p3_higher.inp	Template file for parametric study using M-K analysis with starting point of $\alpha = 0.3$ and higher equivalent plastic strain.
damage_mk_p0p3_higher.psf	Script file for parametric study using M-K analysis with starting point of $\alpha = 0.3$ and higher equivalent plastic strain.
damage_mk_m0p6_lower.inp	Template file for parametric study using M-K analysis with starting point of $\alpha = -0.6$ and lower equivalent plastic strain.
damage_mk_m0p6_lower.psf	Script file for parametric study using M-K analysis with starting point of $\alpha = -0.6$ and lower equivalent plastic strain.
damage_mk_m0p6_higher.inp	Template file for parametric study using M-K analysis with starting point of $\alpha = -0.6$ and higher equivalent plastic strain.
damage_mk_m0p6_higher.psf	Script file for parametric study using M-K analysis with starting point of $\alpha = -0.6$ and higher equivalent plastic strain.
damage_mk_m0p4.inp	Template file for parametric study using M-K analysis with starting point of $\alpha = -0.4$.
damage_mk_m0p4.psf	Script file for parametric study using M-K analysis with starting point of $\alpha = -0.4$.

Figures

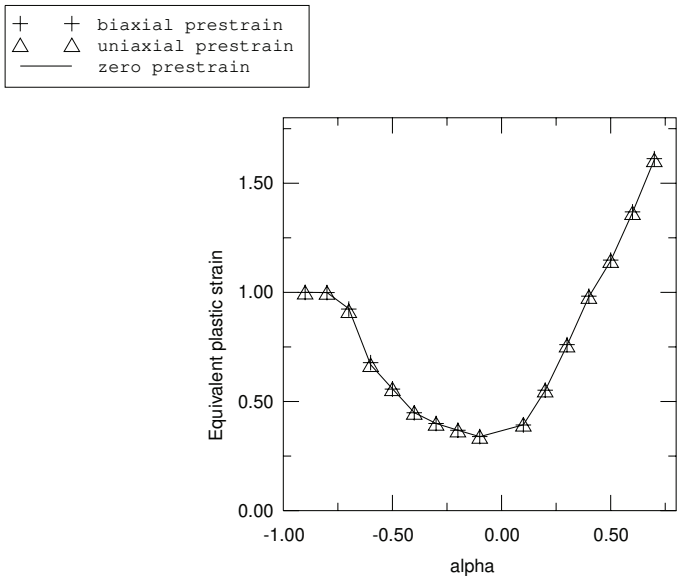


Figure 2: Forming limit diagrams predicted with M-K analyses and plotted in the space of equivalent plastic strain versus ratio of principal strain rates (MSFLD representation).

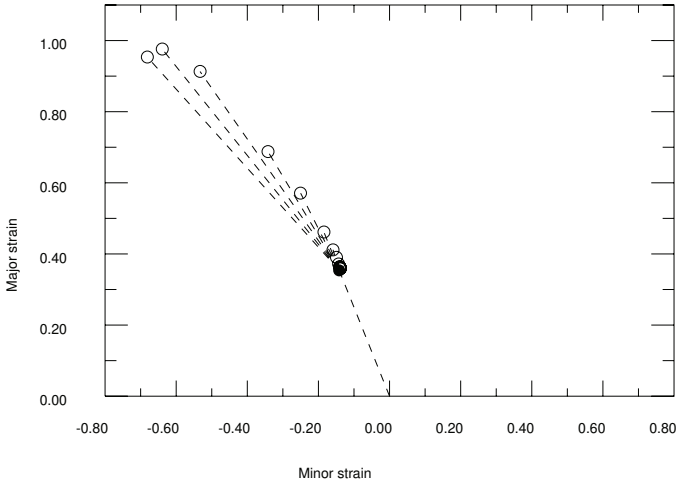


Figure 3: Forming limits predicted using M-K analyses for two-step forming processes with starting point of α = –0.4.

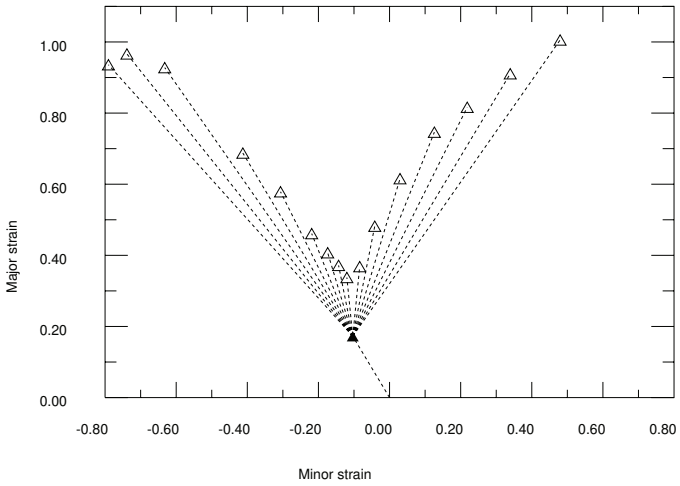


Figure 4: Forming limits predicted using M-K analyses for two-step forming processes with starting point of α = –0.6 and lower equivalent plastic strain.

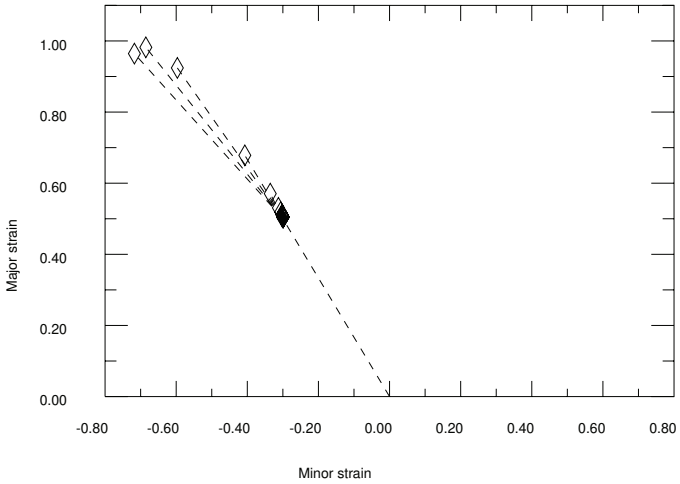


Figure 5: Forming limits predicted using M-K analyses for two-step forming processes with starting point of α = –0.6 and higher equivalent plastic strain.

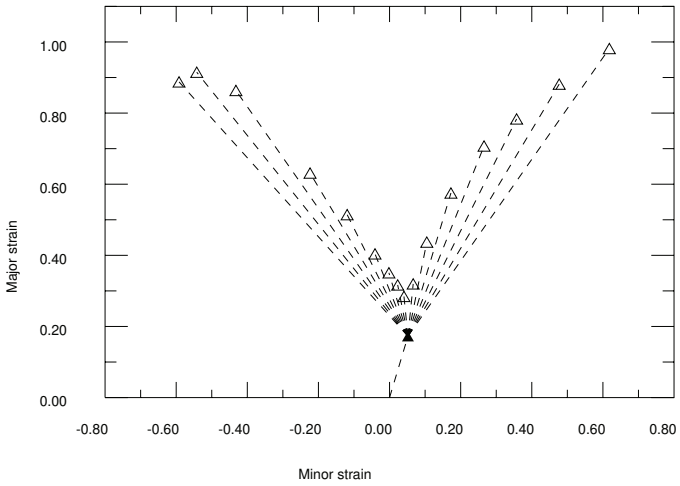


Figure 6: Forming limits predicted using M-K analyses for two-step forming processes with starting point of α = 0.3 and lower equivalent plastic strain.

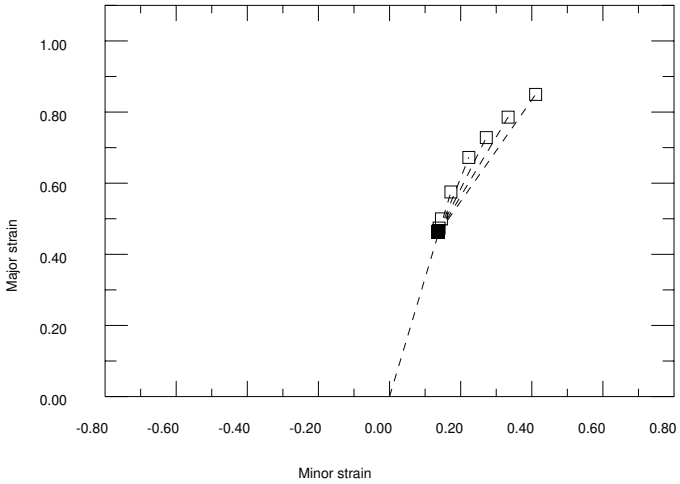


Figure 7: Forming limits predicted using M-K analyses for two-step forming processes with starting point of α = 0.3 and higher equivalent plastic strain.

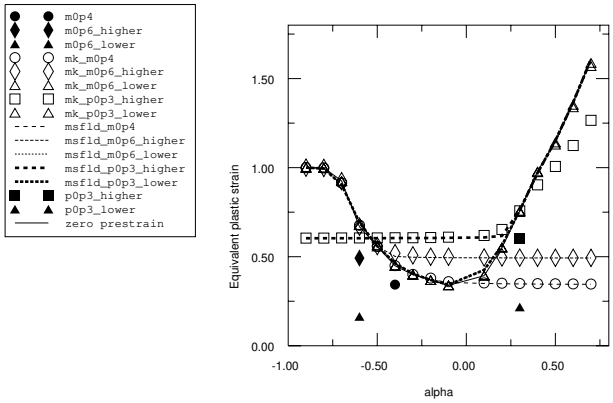


Figure 8: Comparison of forming limit diagrams predicted using MSFLD criterion and those using M-K analyses. (Solid symbols: state at end of first step for various type of loading. Hollow

symbols: state corresponding to initiation of necking during the second step predicted using the M-K analyses. Dashed lines: necking points obtained using the MSFLD criterion. Refer to the input file descriptions for an explanation of the labels.)

Element deletion

Elements tested

T2D2 T3D2 C3D8 C3D8R CPE4R CAX4R

Features tested

The nondefault degradation behavior is tested in Abaqus/Explicit by specifying that fully damaged elements should remain in the computations.

Problem description

The ductile initiation criterion is used on a set of single-element models, subjected to plane strain compression followed by plane strain tension for the elements with two-dimensional and three-dimensional stress states. The truss elements are loaded in uniaxial compression followed by uniaxial tension.

Results and discussion

For elements with two-dimensional and three-dimensional stress states, only the deviatoric and tensile hydrostatic response of the material are degraded once the damage initiation criterion is met; the compressive hydrostatic response is not degraded. For elements with one-dimensional stress states, the stress component is degraded only when it is positive. All elements remain active when element deletion is not used.

Input files

damage_section_no.inp	Nondefault element degradation behavior.

Damage evolution

Elements tested

S4R

Features tested

The maximum and multiplicative rules for computing the overall damage variable from each individual damage variable contribution are tested in Abaqus/Explicit. The field and temperature dependence of the damage initiation criteria and the damage evolution rules are also tested.

Problem description

This verification test consists of six elements, each associated with a different material. For each of the first five materials, only one initiation criterion with its corresponding evolution rule is specified; for the material assigned to the sixth element, all five initiation criteria with their corresponding evolution rules are specified. In this way the individual contribution to the overall damage variable (in the sixth element) can be obtained explicitly from the damage variables of the first five elements.

Results and discussion

The overall damage variable matches with the total contributions from each of the individual damage variables according to the specified combination rule; i.e., maximum or multiplicative.

Input files

damage_combine_deg.inp	Maximum or multiplicative combination rules.

References

- Hill, R., “On Discontinuous Plastic States, with Special Reference to Localized Necking in Thin Sheets,” *Journal of the Mechanics and Physics of Solids*, vol. 1, pp. 19–30, 1952.

Progressive damage and failure in fiber-reinforced materials

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Damage initiation and damage evolution (Hashin criteria)

Elements tested

CPS3 CPS4 CPS4I CPS4R CPS6 CPS6M CPS8 CPS8R
M3D3 M3D4 M3D4R M3D6 M3D8 M3D8R M3D9 M3D9R
S3 S3R S3RS S4 S4R S4R5 S4RS S4RSW S8R S8R5 S9R5 SC6R SC8R STRI3 STRI65

Features tested

The Hashin damage initiation criteria and energy-based damage evolution law are tested with a linearly elastic material.

Problem description

This verification test consists of a set of one- and two-element models subjected to uniaxial tension or compression for various angles (off-axis angles) between the fiber direction and the direction in which the load is applied. The default maximum degradation (equal to 1.0) is used for first-order elements, and the value of the maximum degradation of 0.95 was specified for the second-order elements.

Results and discussion

The degradation of the material stiffness starts when the Hashin initiation criterion is reached for at least one of the failure modes. The damage variables, for the damage modes for which the initiation criteria are satisfied, evolve according to an energy-based evolution law with linear softening. Once the damage variable reaches the maximum degradation specified, no further damage takes place.

The results for the off-axis angles equal to 0° (fiber tension and compression) and 90° (matrix tension and compression) were verified to agree with analytical results.

Figure 1 and *Figure 2* show the unidirectional stress for tension and compression, respectively, at which the initiation criterion is satisfied as a function of the off-axis angle. In these figures the numerical predictions agree very well with the analytical results and also show good agreement with the experimental data reported in *Jones (1999)*.

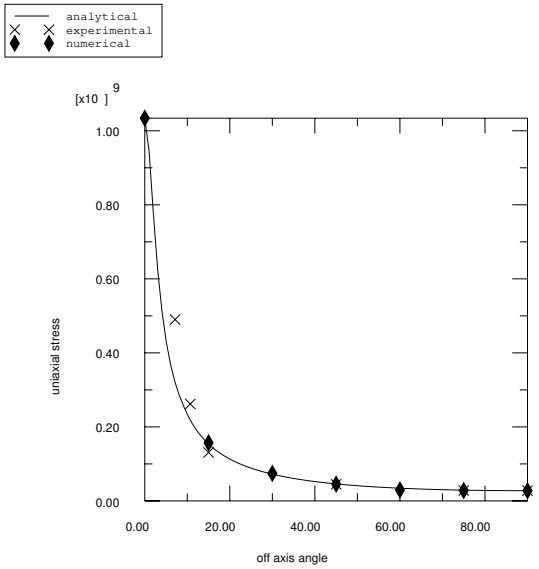


Figure 1: Failure criteria for uniaxial tension as a function of off-axis angle.

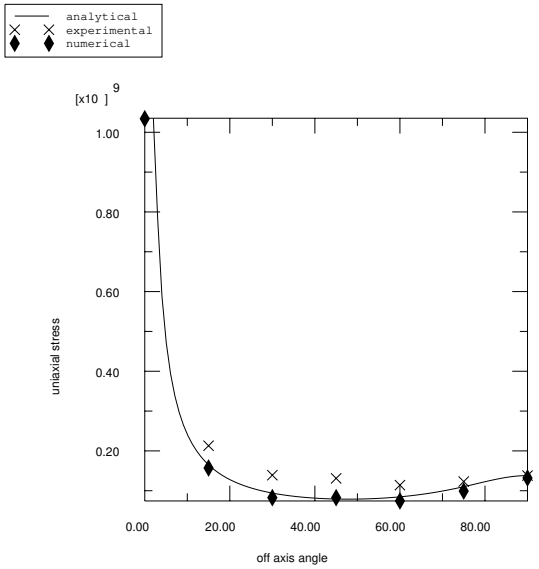


Figure 2: Failure criteria for uniaxial compression as a function of off-axis angle.

Input files

Abaqus/Standard input files

damage_hsncomp_cps4r_0.inp	CPS4 elements are subjected to uniaxial compression; off-axis angle, 0°.
damage_hsncomp_cps4r_90.inp	CPS4R elements are subjected to uniaxial compression; off-axis angle, 90°.
damage_hsncomp_cps6_90.inp	CPS6 elements are subjected to uniaxial compression; off-axis angle, 90°.
damage_hsncomp_cps6m_0.inp	CPS6M elements are subjected to uniaxial compression; off-axis angle, 0°.

damage_hsncomp_cps8_0.inp	CPS8 elements are subjected to uniaxial compression; off-axis angle, 0°.
damage_hsncomp_cps8r_0.inp	CPS8R elements are subjected to uniaxial compression; off-axis angle, 0°.
damage_hsncomp_m3d8_0.inp	M3D8 elements are subjected to uniaxial compression; off-axis angle, 0°.
damage_hsncomp_m3d8r_0.inp	M3D8R elements are subjected to uniaxial compression; off-axis angle, 0°.
damage_hsncomp_m3d9_0.inp	M3D9 elements are subjected to uniaxial compression; off-axis angle, 0°.
damage_hsncomp_s4r_0.inp	S4R elements are subjected to uniaxial compression; off-axis angle, 0°.
damage_hsncomp_s4r_15.inp	S4R elements are subjected to uniaxial compression; off-axis angle, 15°.
damage_hsncomp_s4r_30.inp	S4R elements are subjected to uniaxial compression; off-axis angle, 30°.
damage_hsncomp_s4r_45.inp	S4R elements are subjected to uniaxial compression; off-axis angle, 45°.
damage_hsncomp_s4r_60.inp	S4R elements are subjected to uniaxial compression; off-axis angle, 60°.
damage_hsncomp_s4r_75.inp	S4R elements are subjected to uniaxial compression; off-axis angle, 75°.
damage_hsncomp_s4r_90.inp	S4R elements are subjected to uniaxial compression; off-axis angle, 90°.
damage_hsncomp_s8r_0.inp	S8R elements are subjected to uniaxial compression; off-axis angle, 0°.
damage_hsncomp_s8r5_0.inp	S8R5 elements are subjected to uniaxial compression; off-axis angle, 0°.
damage_hsncomp_s9r5_0.inp	S9R5 elements are subjected to uniaxial compression; off-axis angle, 0°.
damage_hsncomp_sc6r_0.inp	SC6R elements are subjected to uniaxial compression; off-axis angle, 0°.
damage_hsnten_cps3_90.inp	CPS3 elements are subjected to uniaxial tension; off-axis angle, 90°.
damage_hsnten_cps4_30.inp	CPS4 elements are subjected to uniaxial tension; off-axis angle, 30°.
damage_hsnten_cps4i_60.inp	CPS4I elements are subjected to uniaxial tension; off-axis angle, 60°.
damage_hsnten_cps4r_0.inp	CPS4R elements are subjected to uniaxial tension; off-axis angle, 0°.
damage_hsnten_cps4r_90.inp	CPS4R elements are subjected to uniaxial tension; off-axis angle, 90°.
damage_hsnten_m3d3_90.inp	M3D3 elements are subjected to uniaxial tension; off-axis angle, 90°.
damage_hsnten_m3d4r_0.inp	M3D4R elements are subjected to uniaxial tension; off-axis angle, 0°.
damage_hsnten_m3d6_90.inp	M3D6 elements are subjected to uniaxial tension; off-axis angle, 90°.

damage_hsnten_m3d9r_0.inp	M3D9R elements are subjected to uniaxial tension; off-axis angle, 0°.
damage_hsnten_s3_0.inp	S3 elements are subjected to uniaxial tension; off-axis angle, 0°.
damage_hsnten_s3r_90.inp	S3R elements are subjected to uniaxial tension; off-axis angle, 90°.
damage_hsnten_s4_90.inp	S4 elements are subjected to uniaxial tension; off-axis angle, 90°.
damage_hsnten_s4r_0.inp	S4R elements are subjected to uniaxial tension; off-axis angle, 0°.
damage_hsnten_s4r_15.inp	S4R elements are subjected to uniaxial tension; off-axis angle, 15°.
damage_hsnten_s4r_30.inp	S4R elements are subjected to uniaxial tension; off-axis angle, 30°.
damage_hsnten_s4r_45.inp	S4R elements are subjected to uniaxial tension; off-axis angle, 45°.
damage_hsnten_s4r_60.inp	S4R elements are subjected to uniaxial tension; off-axis angle, 60°.
damage_hsnten_s4r_75.inp	S4R elements are subjected to uniaxial tension; off-axis angle, 75°.
damage_hsnten_s4r_90.inp	S4R elements are subjected to uniaxial tension; off-axis angle, 90°.
damage_hsnten_s4r5_90.inp	S4R5 elements are subjected to uniaxial tension; off-axis angle, 90°.
damage_hsnten_sc8r_0.inp	SC8R elements are subjected to uniaxial tension; off-axis angle, 0°.
damage_hsnten_stri3_0.inp	STRI3 elements are subjected to uniaxial tension; off-axis angle, 0°.
damage_hsnten_stri65_90.inp	STRI65 elements subjected to uniaxial tension; off-axis angle, 90°.

Abaqus/Explicit input files

x_damage_hsnten_cps3_45.inp	CPS3 elements are subjected to uniaxial tension; off-axis angle, 45°.
x_damage_hsncomp_cps3_45.inp	CPS3 elements are subjected to uniaxial compression; off-axis angle, 45°.
x_damage_hsnten_cps4r_45.inp	CPS4R elements are subjected to uniaxial tension; off-axis angle, 45°.
x_damage_hsncomp_cps4r_45.inp	CPS4R elements are subjected to uniaxial compression; off-axis angle, 45°.
x_damage_hsnten_m3d3_45.inp	M3D3 elements are subjected to uniaxial tension; off-axis angle, 45°.
x_damage_hsncomp_m3d3_45.inp	M3D3 elements are subjected to uniaxial compression; off-axis angle, 45°.
x_damage_hsnten_m3d4r_45.inp	M3D4R elements are subjected to uniaxial tension; off-axis angle, 45°.
x_damage_hsncomp_m3d4r_45.inp	M3D4R elements are subjected to uniaxial compression; off-axis angle, 45°.
x_damage_hsnten_m3d4_45.inp	M3D4 elements are subjected to uniaxial tension; off-axis angle, 45°.
x_damage_hsncomp_m3d4_45.inp	M3D4 elements are subjected to uniaxial compression; off-axis angle, 45°.
x_damage_hsnten_sc6r_45.inp	SC6R elements are subjected to uniaxial tension; off-axis angle, 45°.
x_damage_hsncomp_sc6r_45.inp	SC6R elements are subjected to uniaxial compression; off-axis angle, 45°.

x_damage_hsnten_sc8r_45.inp	SC8R elements are subjected to uniaxial tension; off-axis angle, 45°.
x_damage_hsncomp_sc8r_45.inp	SC8R elements are subjected to uniaxial compression; off-axis angle, 45°.
x_damage_hsnten_s3_45.inp	S3 elements are subjected to uniaxial tension; off-axis angle, 45°.
x_damage_hsncomp_s3_45.inp	S3 elements are subjected to uniaxial compression; off-axis angle, 45°.
x_damage_hsnten_s3r_45.inp	S3R elements are subjected to uniaxial tension; off-axis angle, 45°.
x_damage_hsncomp_s3r_45.inp	S3R elements are subjected to uniaxial compression; off-axis angle, 45°.
x_damage_hsnten_s4_45.inp	S4 elements are subjected to uniaxial tension; off-axis angle, 45°.
x_damage_hsncomp_s4_45.inp	S4 elements are subjected to uniaxial compression; off-axis angle, 45°.
x_damage_hsnten_s4r_0.inp	S4R elements are subjected to uniaxial tension; off-axis angle, 0°.
x_damage_hsnten_s4r_15.inp	S4R elements are subjected to uniaxial tension; off-axis angle, 15°.
x_damage_hsnten_s4r_30.inp	S4R elements are subjected to uniaxial tension; off-axis angle, 30°.
x_damage_hsnten_s4r_45.inp	S4R elements are subjected to uniaxial tension; off-axis angle, 45°.
x_damage_hsnten_s4r_60.inp	S4R elements are subjected to uniaxial tension; off-axis angle, 60°.
x_damage_hsnten_s4r_75.inp	S4R elements are subjected to uniaxial tension; off-axis angle, 75°.
x_damage_hsnten_s4r_90.inp	S4R elements are subjected to uniaxial tension; off-axis angle, 90°.
x_damage_hsncomp_s4r_0.inp	S4R elements are subjected to uniaxial compression; off-axis angle, 0°.
x_damage_hsncomp_s4r_15.inp	S4R elements are subjected to uniaxial compression; off-axis angle, 15°.
x_damage_hsncomp_s4r_30.inp	S4R elements are subjected to uniaxial compression; off-axis angle, 30°.
x_damage_hsncomp_s4r_45.inp	S4R elements are subjected to uniaxial compression; off-axis angle, 45°.
x_damage_hsncomp_s4r_60.inp	S4R elements are subjected to uniaxial compression; off-axis angle, 60°.
x_damage_hsncomp_s4r_75.inp	S4R elements are subjected to uniaxial compression; off-axis angle, 75°.
x_damage_hsncomp_s4r_90.inp	S4R elements are subjected to uniaxial compression; off-axis angle, 90°.

References

- Jones, R. M., “Mechanics of Composite Materials,” *Taylor & Francis, Inc.*, pp. 102–112, 1999.

LaRC05 criterion

Elements tested

C3D4 C3D6 C3D8 C3D8R C3D10 CPS4 CPS4R
M3D4 M3D4R
S3 S3R S4 S8R S8R5 STRI65

Features tested

The LaRC05 damage initiation criterion is tested with a linearly elastic material.

Problem description

This verification model is a long plate with a circle in the center subjected to uniaxial tension, as shown in *Figure 3*. The LaRC05 damage criterion is tested for this model with a number of different element types. The criterion is also used in conjunction with damage evolution for a similar model to simulate crack nucleation and subsequent propagation in an extended finite element method (XFEM) analysis. The optional parameter FAILURE INDEX must have been specified for **DAMAGE EVOLUTION* in this case. FAILURE INDEX = 1, 2, 3, and 4 correspond to matrix cracking, fiber kinking, fiber splitting, and fiber tension, respectively.

Results and discussion

If you define the damage initiation criterion without defining an associated damage evolution law, the initiation criterion affects only output (the material response is not affected). This is the case for most of the models presented here. The damage initiation criterion output variables LARCMCCRT, LARCFKCRT, LARCFSCRT, and LARCFTCRT were reviewed. The numerical results match the results obtained by hand calculations, based on the formula presented in *LaRC05 Criterion*. *Figure 4* shows the crack nucleation and propagation in an XFEM analysis with three different scenarios (that is, off-axis angles equal to 0°, -45°, and 90°).

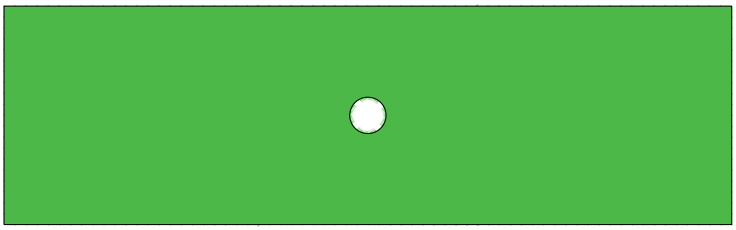


Figure 3: LaRC05 criterion for uniaxial tension.

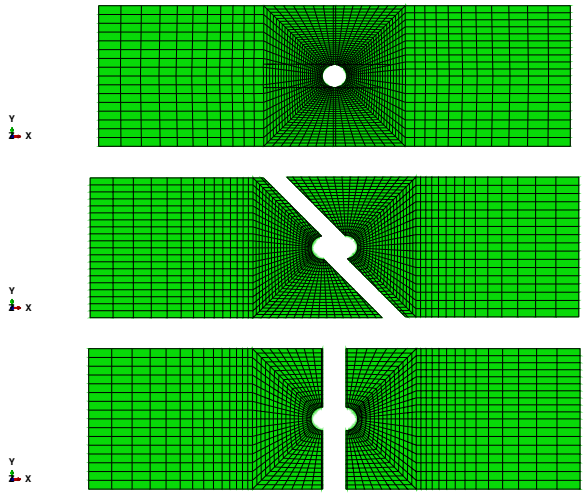


Figure 4: LaRC05 criterion for uniaxial tension in XFEM.

Input files

larc05_dmgcrt_c3d10.inp	C3D10 elements are subjected to uniaxial tension.
larc05_dmgcrt_c3d4.inp	C3D4 elements are subjected to uniaxial tension.
larc05_dmgcrt_c3d6.inp	C3D6 elements are subjected to uniaxial tension.
larc05_dmgcrt_cps4.inp	CPS4 elements are subjected to uniaxial tension.
larc05_dmgcrt_cps4r.inp	CPS4R elements are subjected to uniaxial tension.
larc05_dmgcrt_m3d4.inp	M3D4 elements are subjected to uniaxial tension.
larc05_dmgcrt_m3d4r.inp	M3D4R elements are subjected to uniaxial tension.
larc05_dmgcrt_s3.inp	S3 elements are subjected to uniaxial tension.
larc05_dmgcrt_s3r.inp	S3R elements are subjected to uniaxial tension.
larc05_dmgcrt_s4.inp	S4 elements are subjected to uniaxial tension.
larc05_dmgcrt_s4r.inp	S4R elements are subjected to uniaxial tension.
larc05_dmgcrt_s4r5.inp	S4R5 elements are subjected to uniaxial tension.
larc05_dmgcrt_s8r.inp	S8R elements are subjected to uniaxial tension.
larc05_dmgcrt_s8r5.inp	S8R5 elements are subjected to uniaxial tension.
larc05_dmgcrt_stri65.inp	STRI65 elements are subjected to uniaxial tension.
larc05_dmgini_00.inp	3D XFEM model is subjected to uniaxial tension, off-axis angle, 0°.
larc05_dmgini_45.inp	3D XFEM model is subjected to uniaxial tension, off-axis angle, 45°.
larc05_dmgini_90.inp	3D XFEM model is subjected to uniaxial tension, off-axis angle, 90°.

Ply fabric criterion

Elements tested

CPS4R S4R

Features tested

The ply fabric damage initiation criterion with bilamina elasticity and shear plasticity is tested.

Problem description

This verification problem contains one-element models subjected to uniaxial tension in the vertical direction with or without compression in the horizontal direction. The ply fabric damage initiation criterion is tested for this model with S4R and CPS4R elements. The criterion is also tested in conjunction with damage evolution.

Results and discussion

The models without compression in the horizontal direction exhibit an elastic response that is consistent with the tensile moduli of the bilamina elasticity definition. When the damage initiation criterion is defined without an associated damage evolution law, the initiation criterion affects only output (the material response is not affected). The results for the damage initiation criterion output variables PLF1TCRT, PLF1CCRT, PLF2TCRT, PLF2CCRT, and PLSHRCRT are consistent with the expected solution. All numerical results match the solutions obtained by hand calculations, based on the formula presented in *Ply Fabric Criterion*.

Input files

Abaqus/Standard input files

ply_fabric_s2t.inp	S4R element subjected to uniaxial tension with bilamina elasticity, shear plasticity, damage initiation, and evolution.
ply_fabric_s2t_cps4.inp	CPS4R element subjected to uniaxial tension with bilamina elasticity, shear plasticity, damage initiation, and evolution.
ply_fabric_s2ts1c_crt.inp	S4R element subjected to combined tension and compression without damage evolution.
ply_fabric_s2ts1c_Ela.inp	S4R element subjected to combined tension and compression with bilamina elasticity only.
ply_fabric_shear_plasYesDmgNo_std.inp	S4R element subjected to combined tension and compression with bilamina elasticity and shear plasticity only.
ply_fabric_shear_rate_std.inp	S4R element subjected to combined tension and compression with power law rate-dependent shear plasticity.
ply_fabric_shear_rateJC_std.inp	S4R element subjected to combined tension and compression with Johnson-Cook rate-dependent shear plasticity.
ply_fabric_shear_rateJC_tab_std.inp	S4R element subjected to combined tension and compression with Johnson-Cook rate-dependent shear plasticity with tabulated hardening data.

Abaqus/Explicit input files

ply_fabric_x2t.inp	S4R element subjected to uniaxial tension with bilamina elasticity, shear plasticity, damage initiation, and evolution.
ply_fabric_x2t_cps4.inp	CPS4R element subjected to uniaxial tension with bilamina elasticity, shear plasticity, damage initiation, and evolution.
ply_fabric_x2tx1c_crt.inp	S4R element subjected to combined tension and compression without damage evolution.
ply_fabric_x2tx1c_Ela.inp	S4R element subjected to combined tension and compression with bilamina elasticity only.

ply_fabric_shear_plasYesDmgNo.inp	S4R element subjected to combined tension and compression with bilamina elasticity and shear plasticity only.
ply_fabric_shear_rate.inp	S4R element subjected to combined tension and compression with power law rate-dependent shear plasticity.
ply_fabric_shear_rateJC.inp	S4R element subjected to combined tension and compression with Johnson-Cook rate-dependent shear plasticity.
ply_fabric_shear_rateJC_tab.inp	S4R element subjected to combined tension and compression with Johnson-Cook rate-dependent shear plasticity with tabulated hardening data.

Multiscale criteria (compared with Hashin criteria)

Elements tested

S4R

Features tested

The multiscale damage initiation criteria and discontinuous damage evolution law are tested with a linearly elastic material.

Problem description

This verification test consists of a set of one-element models subjected to uniaxial tension or compression for various angles (off-axis angles) between the fiber direction and the direction in which the load is applied. The failure strengths of the composite material are set to be the same as those used in *Damage initiation and damage evolution (Hashin criteria)*. The elastic constants of the constituents are calibrated using the elastic constants specified for the composite in *Damage initiation and damage evolution (Hashin criteria)*. The HSNFIBER criterion is specified for the fiber material, and the HSNMATRIX criterion is specified for the matrix material.

Multiscale damage initiation criterion and evolution results are also compared to those using the Hashin model in an open-hole tension test. The model is the same as the S4R model used in *LaRC05 criterion*. The maximum degradation is set to 0.99 for the Hashin model and for the fiber and matrix in the multiscale model. For better comparison, *UDAMAGEMF* is used to degrade the matrix stiffness on fiber damage initiation.

Results and discussion

The degradation of the material stiffness starts when the initiation criterion is reached for at least one of the constituents. For simplicity, discontinuous damage evolution is specified, and the maximum degradation specified by the user is applied on damage initiation. The maximum degradation is specified in the damage evolution law for each constituent. The degradation is set to 0.85 for the matrix material and 0.99 for the fiber material.

Figure 5 and *Figure 6* show the unidirectional stress for tension and compression, respectively, at which the initiation criterion is satisfied as a function of the off-axis angle. In these figures, the numerical predictions agree very well with the analytical results shown in *Figure 1* and *Figure 2*. The predictions also show good agreement with the experimental data reported in *Jones (1999)*.

Figure 7 shows the comparison of reaction force versus displacement in an open-hole tension test. The multiscale results predict the same ultimate strength of the plate as the Hashin results at various off-axis angles.

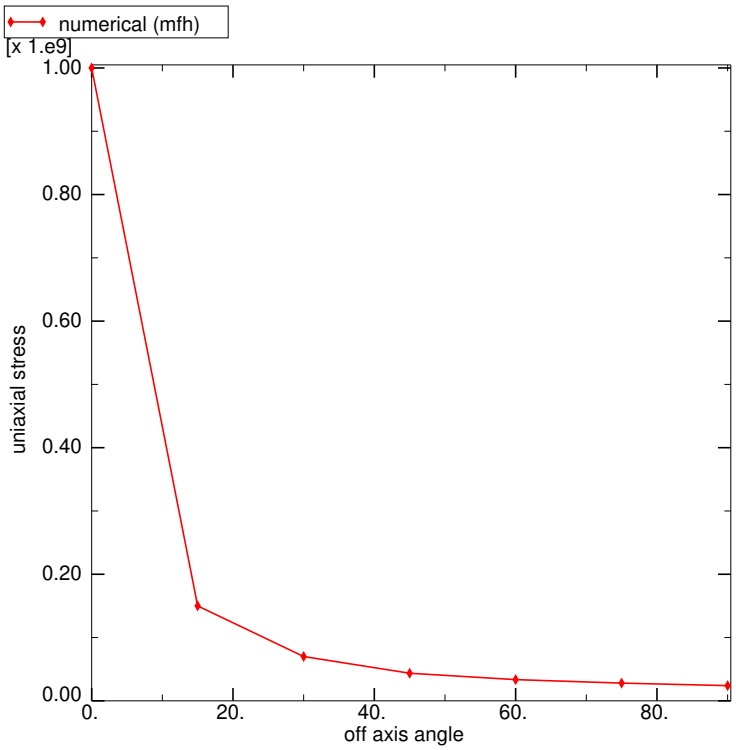


Figure 5: Failure criteria for uniaxial tension as a function of off-axis angle.

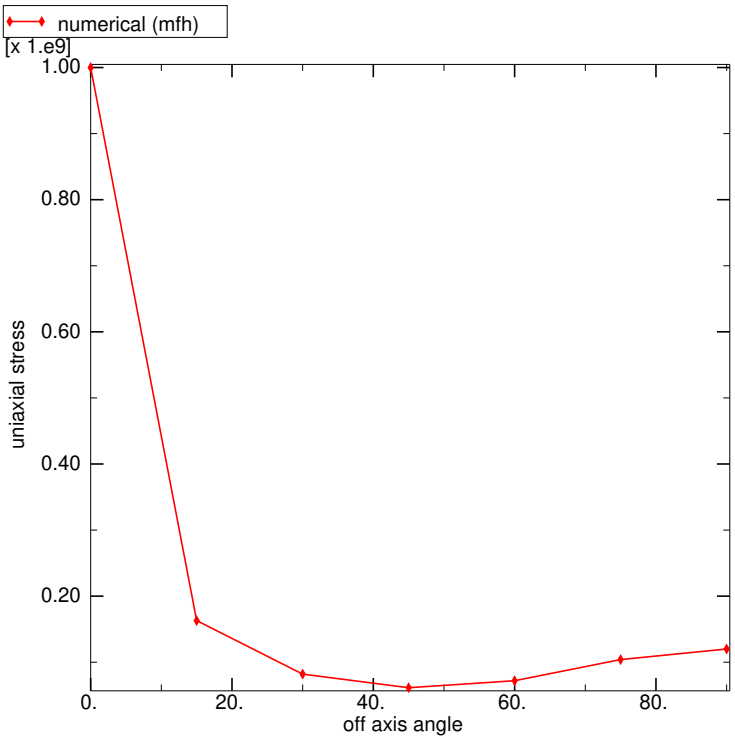
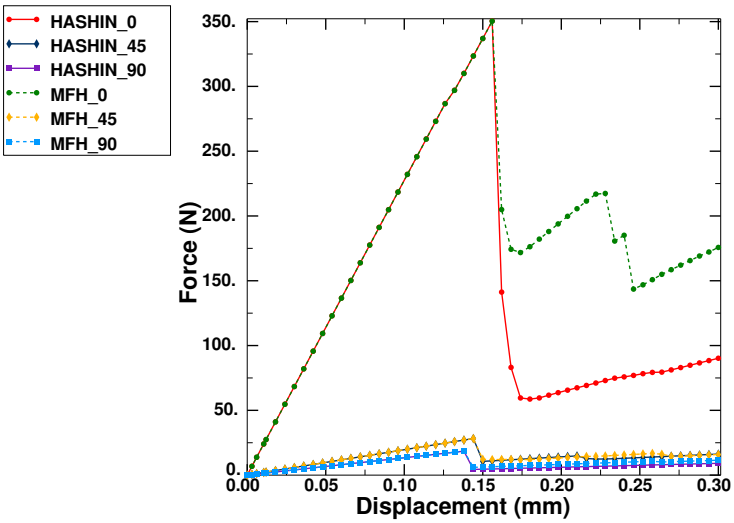


Figure 6: Failure criteria for uniaxial compression as a function of off-axis angle.

Figure 7: Reaction force comparison between multiscale and Hashin model at various off-axis angles.



Input files

Abaqus/Standard input files

damage_mfhcomp_s4r_0.inp	S4R elements subjected to uniaxial compression; off-axis angle, 0°.
damage_mfhcomp_s4r_15.inp	S4R elements subjected to uniaxial compression; off-axis angle, 15°.
damage_mfhcomp_s4r_30.inp	S4R elements subjected to uniaxial compression; off-axis angle, 30°.
damage_mfhcomp_s4r_45.inp	S4R elements subjected to uniaxial compression; off-axis angle, 45°.
damage_mfhcomp_s4r_60.inp	S4R elements subjected to uniaxial compression; off-axis angle, 60°.
damage_mfhcomp_s4r_75.inp	S4R elements subjected to uniaxial compression; off-axis angle, 75°.
damage_mfhcomp_s4r_90.inp	S4R elements subjected to uniaxial compression; off-axis angle, 90°.
damage_mfhten_s4r_0.inp	S4R elements subjected to uniaxial tension; off-axis angle, 0°.
damage_mfhten_s4r_15.inp	S4R elements subjected to uniaxial tension; off-axis angle, 15°.
damage_mfhten_s4r_30.inp	S4R elements subjected to uniaxial tension; off-axis angle, 30°.
damage_mfhten_s4r_45.inp	S4R elements subjected to uniaxial tension; off-axis angle, 45°.
damage_mfhten_s4r_60.inp	S4R elements subjected to uniaxial tension; off-axis angle, 60°.
damage_mfhten_s4r_75.inp	S4R elements subjected to uniaxial tension; off-axis angle, 75°.
damage_mfhten_s4r_90.inp	S4R elements subjected to uniaxial tension; off-axis angle, 90°.
hashin_dmgini_s4r_0.inp	S4R elements in a plate model subjected to uniaxial tension; off-axis angle, 0°. Hashin damage model.
hashin_dmgini_s4r_45.inp	S4R elements in a plate model subjected to uniaxial tension; off-axis angle, 45°. Hashin damage model.
hashin_dmgini_s4r_90.inp	S4R elements in a plate model subjected to uniaxial tension; off-axis angle, 90°. Hashin damage model.
mfh_dmgini_s4r_0.inp	S4R elements in a plate model subjected to uniaxial tension; off-axis angle, 0°. Multiscale damage model.
mfh_dmgini_s4r_45.inp	S4R elements in a plate model subjected to uniaxial tension; off-axis angle, 45°. Multiscale damage model.
mfh_dmgini_s4r_90.inp	S4R elements in a plate model subjected to uniaxial tension; off-axis angle, 90°. Multiscale damage model.
udamagemf_ver.f	User subroutine *UDAMAGEMF* for post–fiber failure degradation and additional SDV output to show the damage mode (1- no damage, 2- matrix damage, 3- fiber damage).

Multiscale criteria (compared with LaRC05 criterion)

Elements tested

C3D8R C3D10 CPS4 M3D4 S4R S4R5 S8R5 SC8R STRI65

Features tested

The multiscale damage initiation criteria and discontinuous damage evolution law are tested with a linearly elastic material.

Problem description

This verification model is the same plate model used in *LaRC05 criterion*. The micro-level damage initiation criterion HSNFIBER and HSNMATRIX are tested for this model with a number of different element types.

Elastic constants of constituents are calibrated using composite properties specified for the material used in *LaRC05 criterion*. The criterion is also used in conjunction with discontinuous damage evolution for a similar model to simulate progressive damage in a static analysis. The maximum degradation of the matrix is set to 0.9, and the maximum degradation of the fiber is set to 0.99.

Results and discussion

If you define the damage initiation criterion without defining an associated damage evolution law, the initiation criterion affects only output (the material response is not affected). This is the case for most of the models presented here. The damage initiation criterion output variables HSNFTCRT, HSNFCCRT, HSNMTCRT, and HSNMCCRT were reviewed. The numerical results match the results obtained by hand calculations, based on the formula presented in *Damage Initiation for Fiber-Reinforced Composites Using Multiscale Modeling*. The damage initiation output variable HSNFCCRT is also compared with LARCFSCRT from the results in *LaRC05 criterion* using C3D10 elements, as shown in *Figure 8*. The damage locations match well.

Figure 9 shows the matrix damage variable in a uniaxial tension model with three different scenarios (that is, off-axis angles equal to 0°, −45°, and 90°). The results compare well with the 3D XFEM results shown in *Figure 4*

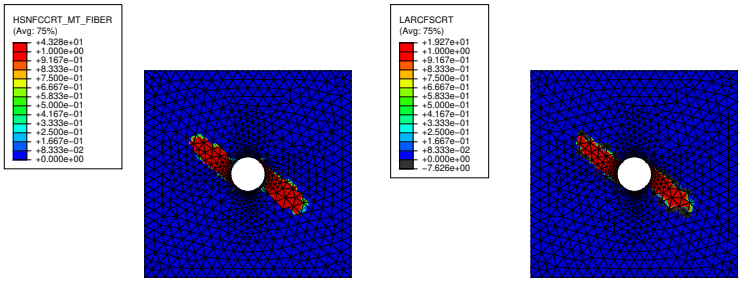


Figure 8: Damage initiation criterion for uniaxial tension.

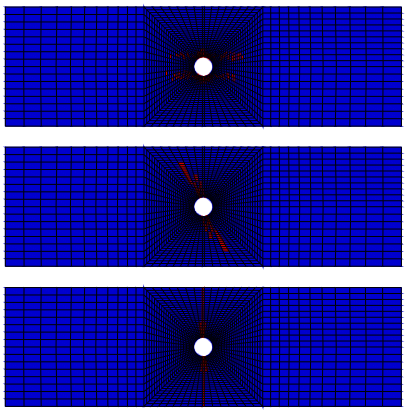


Figure 9: Damage variable for matrix in an open-hole uniaxial tension model.

Input files

Abaqus/Standard input files

mfhdam_dmgcrt_c3d10.inp	C3D10 elements in open-hole plate model subjected to uniaxial tension; off-axis angle, −45°.
mfhdam_dmgcrt_cps4.inp	CPS4 elements in open-hole plate model subjected to uniaxial tension; off-axis angle, −45°.
mfhdam_dmgcrt_m3d4.inp	M3D4 elements in open-hole plate model subjected to uniaxial tension; off-axis angle, −45°.
mfhdam_dmgcrt_s4r.inp	S4R elements in open-hole plate model subjected to uniaxial tension; off-axis angle, −45°.
mfhdam_dmgcrt_s4r5.inp	S4R5 elements in open-hole plate model subjected to uniaxial tension; off-axis angle, −45°.
mfhdam_dmgcrt_s8r5.inp	S8R5 elements in open-hole plate model subjected to uniaxial tension; off-axis angle, −45°.
mfhdam_dmgcrt_stri65.inp	STRI65 elements in open-hole plate model subjected to uniaxial tension; off-axis angle, −45°.
mfhdam_dmgini_00.inp	C3D8R elements in open-hole plate model subjected to uniaxial tension; off-axis angle, 0°.
mfhdam_dmgini_45.inp	C3D8R elements in open-hole plate model subjected to uniaxial tension; off-axis angle, 45°.
mfhdam_dmgini_90.inp	C3D8R elements in open-hole plate model subjected to uniaxial tension; off-axis angle, 90°.

Multiscale criteria (compared with ply fabric criteria)

Elements tested

C3D8R CPS4 S4R SC8R

Features tested

The multiscale damage initiation criteria and discontinuous damage evolution law are tested with a linearly elastic, woven composite material.

Problem description

This verification test consists of a set of one-element models subjected to uniaxial tension or compression load. The failure strengths of the composite material are set to be the same as those used in *Ply fabric criterion*. The elastic constants of the constituents are calibrated using the elastic constants specified for the composite in *Ply fabric criterion*. The HSNFIBER criterion is specified for the fiber material, and the TSINVMATRIX criterion is specified for the matrix material.

Two-level homogenization is used in the multiscale material model. Properties are specified for the fiber and matrix material in the tows. The tow material is then used as the inclusion material for the woven composite. The matrix material between the tows is assumed to have the same properties as the matrix material in the tows, with the exception of damage. Considering the damage in the matrix pockets (matrix between tows) does not have significant effect on the failure strength of the composite; therefore, the damage in the matrix pocket is ignored.

Results and discussion

The degradation of the material stiffness starts when the initiation criterion is reached for at least one of the constituents. For simplicity, discontinuous damage evolution is specified, and the maximum degradation specified by the user is applied on damage initiation. The maximum degradation is specified in the damage evolution law for each constituent. The degradation is set to 0.9 for the matrix material and 0.99 for the fiber material.

The results for the damage initiation criterion output variables HSNFTCRT, HSNFCCRT,TSINVMTCRT, and TSINVMCCRT are consistent with the expected solution. All numerical results match the solutions obtained by hand calculations, based on the formula presented in *Damage Initiation for Fiber-Reinforced Composites Using Multiscale Modeling*.

Input files

Python input files

mfh_calib_woven.py	Script used to calibrate the constituent properties for the woven multiscale model.

Abaqus/Standard input files

mfh_woven_x2c_s4r.inp	S4R elements subjected to uniaxial compression.
mfh_woven_x2t_s4r.inp	S4R elements subjected to uniaxial tension.
mfh_woven_x2t_cps4.inp	CPS4 elements subjected to uniaxial tension.
mfh_woven_x2t_sc8r.inp	SC8R elements subjected to uniaxial tension.
mfh_woven_x2t_crt_s4r.inp	S4R elements subjected to uniaxial tension. No damage evolution.
mfh_woven_x2t_crt_cps4.inp	CPS4 elements subjected to uniaxial tension. No damage evolution.

Import from Abaqus/Standard to Abaqus/Explicit (Hashin model)

Elements tested

CPS3 CPS4R
M3D3 M3D4 M3D4R
S3R S4 S4R SC6R SC8R

Problem description

This category of problems tests the import capability from Abaqus/Standard to Abaqus/Explicit with the Hashin damage model. All tests subject the elements to uniaxial tension and compression loading in Abaqus/Standard. The model is then imported into Abaqus/Explicit and is subjected to further uniaxial tension and compression loading. Two fiber orientations, 0° and 45°, are considered. All the tests include problems that import neither the reference configuration nor the state, problems that import only the state, problems that import only the reference configuration, and problems that import both the reference configuration and the state.

Results and discussion

The import capability is validated by comparing various damage variables and energy dissipation due to damage after each import of the results; the response after import is as expected.

Input files

sx_s_dmg_hsntencomp_cps_0.inp	Base problem for carrying out import from Abaqus/Standard to Abaqus/Explicit; CPS3 and CPS4R elements; fiber orientation 0°.
sx_s_dmg_hsntencomp_cps_45.inp	Base problem for carrying out import from Abaqus/Standard to Abaqus/Explicit; CPS3 and CPS4R elements; fiber orientation 45°.
sx_s_dmg_hsntencomp_mem_0.inp	Base problem for carrying out import from Abaqus/Standard to Abaqus/Explicit; M3D3, M3D4R, and M3D4 elements; fiber orientation 0°.
sx_s_dmg_hsntencomp_mem_45.inp	Base problem for carrying out import from Abaqus/Standard to Abaqus/Explicit; M3D3, M3D4R, and M3D4 elements; fiber orientation 45°.
sx_s_dmg_hsntencomp_shell_0.inp	Base problem for carrying out import from Abaqus/Standard to Abaqus/Explicit; S3R, S4R, and S4 elements; fiber orientation 0°.
sx_s_dmg_hsntencomp_shell_45.inp	Base problem for carrying out import from Abaqus/Standard to Abaqus/Explicit; S3R, S4R, and S4 elements; fiber orientation 45°.
sx_s_dmg_hsntencomp_cshell_0.inp	Base problem for carrying out import from Abaqus/Standard to Abaqus/Explicit; SC6R and SC8R elements; fiber orientation 0°.
sx_s_dmg_hsntencomp_cshell_45.inp	Base problem for carrying out import from Abaqus/Standard to Abaqus/Explicit; SC6R and SC8R elements; fiber orientation 45°.
sx_x_dmg_hsntencomp_cps_0_n_n.inp	Explicit dynamic continuation of `sx_s_dmg_hsntencomp_cps_0.inp` without importing the reference configuration or the state; CPS3 and CPS4R elements; fiber orientation 0°.
sx_x_dmg_hsntencomp_cps_0_n_y.inp	Explicit dynamic continuation of `sx_s_dmg_hsntencomp_cps_0.inp` with only the state imported; CPS3 and CPS4R elements; fiber orientation 0°.
sx_x_dmg_hsntencomp_cps_0_y_n.inp	Explicit dynamic continuation of `sx_s_dmg_hsntencomp_cps_0.inp` with only the reference configuration imported; CPS3 and CPS4R elements; fiber orientation 0°.
sx_x_dmg_hsntencomp_cps_0_y_y.inp	Explicit dynamic continuation of `sx_s_dmg_hsntencomp_cps_0.inp` with both the reference configuration and the state imported; CPS3 and CPS4R elements; fiber orientation 0°.
sx_x_dmg_hsntencomp_cps_45_n_n.inp	Explicit dynamic continuation of `sx_s_dmg_hsntencomp_cps_45.inp` without importing the reference configuration or the state; CPS3 and CPS4R elements; fiber orientation 45°.
sx_x_dmg_hsntencomp_cps_45_n_y.inp	Explicit dynamic continuation of `sx_s_dmg_hsntencomp_cps_45.inp` with only the state imported; CPS3 and CPS4R elements; fiber orientation 45°.
sx_x_dmg_hsntencomp_cps_45_y_n.inp	Explicit dynamic continuation of `sx_s_dmg_hsntencomp_cps_45.inp` with only the reference configuration imported; CPS3 and CPS4R elements; fiber orientation 45°.
sx_x_dmg_hsntencomp_cps_45_y_y.inp	Explicit dynamic continuation of `sx_s_dmg_hsntencomp_cps_45.inp` with both the reference configuration and the state imported; CPS3 and CPS4R elements; fiber orientation 45°.

sx_x_dmg_hsntencomp_mem_0_n_n.inp	Explicit dynamic continuation of `sx_s_dmg_hsntencomp_mem_0.inp` without importing the reference configuration or the state; M3D3, M3D4R, and M3D4 elements; fiber orientation 0°.
sx_x_dmg_hsntencomp_mem_0_n_y.inp	Explicit dynamic continuation of `sx_s_dmg_hsntencomp_mem_0.inp` with only the state imported; M3D3, M3D4R, and M3D4 elements; fiber orientation 0°.
sx_x_dmg_hsntencomp_mem_0_y_n.inp	Explicit dynamic continuation of `sx_s_dmg_hsntencomp_mem_0.inp` with only the reference configuration imported; M3D3, M3D4R, and M3D4 elements; fiber orientation 0°.
sx_x_dmg_hsntencomp_mem_0_y_y.inp	Explicit dynamic continuation of `sx_s_dmg_hsntencomp_mem_0.inp` with both the reference configuration and the state imported; M3D3, M3D4R, and M3D4 elements; fiber orientation 0°.
sx_x_dmg_hsntencomp_mem_45_n_n.inp	Explicit dynamic continuation of `sx_s_dmg_hsntencomp_mem_45.inp` without importing the reference configuration or the state; M3D3, M3D4R, and M3D4 elements; fiber orientation 45°.
sx_x_dmg_hsntencomp_mem_45_n_y.inp	Explicit dynamic continuation of `sx_s_dmg_hsntencomp_mem_45.inp` with only the state imported; M3D3, M3D4R, and M3D4 elements; fiber orientation 45°.
sx_x_dmg_hsntencomp_mem_45_y_n.inp	Explicit dynamic continuation of `sx_s_dmg_hsntencomp_mem_45.inp` with only the reference configuration imported; M3D3, M3D4R, and M3D4 elements; fiber orientation 45°.
sx_x_dmg_hsntencomp_mem_45_y_y.inp	Explicit dynamic continuation of `sx_s_dmg_hsntencomp_mem_45.inp` with both the reference configuration and the state imported; M3D3, M3D4R, and M3D4 elements; fiber orientation 45°.
sx_x_dmg_hsntencomp_shell_0_n_n.inp	Explicit dynamic continuation of `sx_s_dmg_hsntencomp_shell_0.inp` without importing the reference configuration or the state; S3R, S4R, and S4 elements; fiber orientation 0°.
sx_x_dmg_hsntencomp_shell_0_n_y.inp	Explicit dynamic continuation of `sx_s_dmg_hsntencomp_shell_0.inp` with only the state imported; S3R, S4R, and S4 elements; fiber orientation 0°.
sx_x_dmg_hsntencomp_shell_0_y_n.inp	Explicit dynamic continuation of `sx_s_dmg_hsntencomp_shell_0.inp` with only the reference configuration imported; S3R, S4R, and S4 elements; fiber orientation 0°.
sx_x_dmg_hsntencomp_shell_0_y_y.inp	Explicit dynamic continuation of `sx_s_dmg_hsntencomp_shell_0.inp` with both the reference configuration and the state imported; S3R, S4R, and S4 elements; fiber orientation 0°.
sx_x_dmg_hsntencomp_shell_45_n_n.inp	Explicit dynamic continuation of `sx_s_dmg_hsntencomp_shell_45.inp` without importing the reference configuration or the state; S3R, S4R, and S4 elements; fiber orientation 45°.

sx_x_dmg_hsntencomp_shell_45_n_y.inp	Explicit dynamic continuation of `sx_s_dmg_hsntencomp_shell_45.inp` with only the state imported; S3R, S4R, and S4 elements; fiber orientation 45°.
sx_x_dmg_hsntencomp_shell_45_y_n.inp	Explicit dynamic continuation of `sx_s_dmg_hsntencomp_shell_45.inp` with only the reference configuration imported; S3R, S4R, and S4 elements; fiber orientation 45°.
sx_x_dmg_hsntencomp_shell_45_y_y.inp	Explicit dynamic continuation of `sx_s_dmg_hsntencomp_shell_45.inp` with both the reference configuration and the state imported; S3R, S4R, and S4 elements; fiber orientation 45°.
sx_x_dmg_hsntencomp_cshell_0_n_n.inp	Explicit dynamic continuation of `sx_s_dmg_hsntencomp_cshell_0.inp` without importing the reference configuration or the state; SC6R and SC8R elements; fiber orientation 0°.
sx_x_dmg_hsntencomp_cshell_0_n_y.inp	Explicit dynamic continuation of `sx_s_dmg_hsntencomp_cshell_0.inp` with only the state imported; SC6R and SC8R elements; fiber orientation 0°.
sx_x_dmg_hsntencomp_cshell_0_y_n.inp	Explicit dynamic continuation of `sx_s_dmg_hsntencomp_cshell_0.inp` with only the reference configuration imported; SC6R and SC8R elements; fiber orientation 0°.
sx_x_dmg_hsntencomp_cshell_0_y_y.inp	Explicit dynamic continuation of `sx_s_dmg_hsntencomp_cshell_0.inp` with both the reference configuration and the state imported; SC6R and SC8R elements; fiber orientation 0°.
sx_x_dmg_hsntencomp_cshell_45_n_n.inp	Explicit dynamic continuation of `sx_s_dmg_hsntencomp_cshell_45.inp` without importing the reference configuration or the state; SC6R and SC8R elements; fiber orientation 45°.
sx_x_dmg_hsntencomp_cshell_45_n_y.inp	Explicit dynamic continuation of `sx_s_dmg_hsntencomp_cshell_45.inp` with only the state imported; SC6R and SC8R elements; fiber orientation 45°.
sx_x_dmg_hsntencomp_cshell_45_y_n.inp	Explicit dynamic continuation of `sx_s_dmg_hsntencomp_cshell_45.inp` with only the reference configuration imported; SC6R and SC8R elements; fiber orientation 45°.
sx_x_dmg_hsntencomp_cshell_45_y_y.inp	Explicit dynamic continuation of `sx_s_dmg_hsntencomp_cshell_45.inp` with both the reference configuration and the state imported; SC6R and SC8R elements; fiber orientation 45°.

Import from Abaqus/Explicit to Abaqus/Standard (Hashin model)

Elements tested

CPS3 CPS4R
M3D3 M3D4 M3D4R
S3R S4 S4R SC6R SC8R

Problem description

This category of problems tests the import capability from Abaqus/Explicit to Abaqus/Standard with the Hashin damage model. All tests subject the elements to uniaxial tension and compression loading in Abaqus/Explicit.

The model is then imported into Abaqus/Standard and is subjected to further uniaxial tension and compression loading. Two fiber orientations, 0° and 45°, are considered. All the tests include problems that import neither the reference configuration nor the state, problems that import only the state, problems that import only the reference configuration, and problems that import both the reference configuration and the state.

Results and discussion

The import capability is validated by comparing various damage variables and energy dissipation due to damage after each import of the results; the response after import is as expected.

Input files

xs_x_dmg_hsntencomp_cps_0.inp	Base problem for carrying out import from Abaqus/Explicit to Abaqus/Standard; CPS3 and CPS4R elements; fiber orientation 0°.
xs_x_dmg_hsntencomp_cps_45.inp	Base problem for carrying out import from Abaqus/Explicit to Abaqus/Standard; CPS3 and CPS4R elements; fiber orientation 45°.
xs_x_dmg_hsntencomp_mem_0.inp	Base problem for carrying out import from Abaqus/Explicit to Abaqus/Standard; M3D3, M3D4R, and M3D4 elements; fiber orientation 0°.
xs_x_dmg_hsntencomp_mem_45.inp	Base problem for carrying out import from Abaqus/Explicit to Abaqus/Standard; M3D3, M3D4R, and M3D4 elements; fiber orientation 45°.
xs_x_dmg_hsntencomp_shell_0.inp	Base problem for carrying out import from Abaqus/Explicit to Abaqus/Standard; S3R, S4R, and S4 elements; fiber orientation 0°.
xs_x_dmg_hsntencomp_shell_45.inp	Base problem for carrying out import from Abaqus/Explicit to Abaqus/Standard; S3R, S4R, and S4 elements; fiber orientation 45°.
xs_x_dmg_hsntencomp_cshell_0.inp	Base problem for carrying out import from Abaqus/Explicit to Abaqus/Standard; SC6R and SC8R elements; fiber orientation 0°.
xs_x_dmg_hsntencomp_cshell_45.inp	Base problem for carrying out import from Abaqus/Explicit to Abaqus/Standard; SC6R and SC8R elements; fiber orientation 45°.
xs_s_dmg_hsntencomp_cps_0_n_n.inp	Static continuation of `xs_x_dmg_hsntencomp_cps_0.inp` without importing the reference configuration or the state; CPS3 and CPS4R elements; fiber orientation 0°.
xs_s_dmg_hsntencomp_cps_0_n_y.inp	Static continuation of `xs_x_dmg_hsntencomp_cps_0.inp` with only the state imported; CPS3 and CPS4R elements; fiber orientation 0°.
xs_s_dmg_hsntencomp_cps_0_y_n.inp	Static continuation of `xs_x_dmg_hsntencomp_cps_0.inp` with only the reference configuration imported; CPS3 and CPS4R elements; fiber orientation 0°.
xs_s_dmg_hsntencomp_cps_0_y_y.inp	Static continuation of `xs_x_dmg_hsntencomp_cps_0.inp` with both the reference configuration and the state imported; CPS3 and CPS4R elements; fiber orientation 0°.
xs_s_dmg_hsntencomp_cps_45_n_n.inp	Static continuation of `xs_x_dmg_hsntencomp_cps_45.inp` without importing the reference configuration or the state; CPS3 and CPS4R elements; fiber orientation 45°.
xs_s_dmg_hsntencomp_cps_45_n_y.inp	Static continuation of `xs_x_dmg_hsntencomp_cps_45.inp` with only the state imported; CPS3 and CPS4R elements; fiber orientation 45°.
xs_s_dmg_hsntencomp_cps_45_y_n.inp	Static continuation of `xs_x_dmg_hsntencomp_cps_45.inp` with only the reference configuration imported; CPS3 and CPS4R elements; fiber orientation 45°.

xs_s_dmg_hsntencomp_cps_45_y_y.inp	Static continuation of `xs_x_dmg_hsntencomp_cps_45.inp` with both the reference configuration and the state imported; CPS3 and CPS4R elements; fiber orientation 45°.
xs_s_dmg_hsntencomp_mem_0_n_n.inp	Static continuation of `xs_x_dmg_hsntencomp_mem_0.inp` without importing the reference configuration or the state; M3D3, M3D4R, and M3D4 elements; fiber orientation 0°.
xs_s_dmg_hsntencomp_mem_0_n_y.inp	Static continuation of `xs_x_dmg_hsntencomp_mem_0.inp` with only the state imported; M3D3, M3D4R, and M3D4 elements; fiber orientation 0°.
xs_s_dmg_hsntencomp_mem_0_y_n.inp	Static continuation of `xs_x_dmg_hsntencomp_mem_0.inp` with only the reference configuration imported; M3D3, M3D4R, and M3D4 elements; fiber orientation 0°.
xs_s_dmg_hsntencomp_mem_0_y_y.inp	Static continuation of `xs_x_dmg_hsntencomp_mem_0.inp` with both the reference configuration and the state imported; M3D3, M3D4R, and M3D4 elements; fiber orientation 0°.
xs_s_dmg_hsntencomp_mem_45_n_n.inp	Static continuation of `xs_x_dmg_hsntencomp_mem_45.inp` without importing the reference configuration or the state; M3D3, M3D4R, and M3D4 elements; fiber orientation 45°.
xs_s_dmg_hsntencomp_mem_45_n_y.inp	Static continuation of `xs_x_dmg_hsntencomp_mem_45.inp` with only the state imported; M3D3, M3D4R, and M3D4 elements; fiber orientation 45°.
xs_s_dmg_hsntencomp_mem_45_y_n.inp	Static continuation of `xs_x_dmg_hsntencomp_mem_45.inp` with only the reference configuration imported; M3D3, M3D4R, and M3D4 elements; fiber orientation 45°.
xs_s_dmg_hsntencomp_mem_45_y_y.inp	Static continuation of `xs_x_dmg_hsntencomp_mem_45.inp` with both the reference configuration and the state imported; M3D3, M3D4R, and M3D4 elements; fiber orientation 45°.
xs_s_dmg_hsntencomp_shell_0_n_n.inp	Static continuation of `xs_x_dmg_hsntencomp_shell_0.inp` without importing the reference configuration or the state; S3R, S4R, and S4 elements; fiber orientation 0°.
xs_s_dmg_hsntencomp_shell_0_n_y.inp	Static continuation of `xs_x_dmg_hsntencomp_shell_0.inp` with only the state imported; S3R, S4R, and S4 elements; fiber orientation 0°.
xs_s_dmg_hsntencomp_shell_0_y_n.inp	Static continuation of `xs_x_dmg_hsntencomp_shell_0.inp` with only the reference configuration imported; S3R, S4R, and S4 elements; fiber orientation 0°.
xs_s_dmg_hsntencomp_shell_0_y_y.inp	Static continuation of `xs_x_dmg_hsntencomp_shell_0.inp` with both the reference configuration and the state imported; S3R, S4R, and S4 elements; fiber orientation 0°.
xs_s_dmg_hsntencomp_shell_45_n_n.inp	Static continuation of `xs_x_dmg_hsntencomp_shell_45.inp` without importing the reference configuration or the state; S3R, S4R, and S4 elements; fiber orientation 45°.
xs_s_dmg_hsntencomp_shell_45_n_y.inp	Static continuation of `xs_x_dmg_hsntencomp_shell_45.inp` with only the state imported; S3R, S4R, and S4 elements; fiber orientation 45°.
xs_s_dmg_hsntencomp_shell_45_y_n.inp	Static continuation of `xs_x_dmg_hsntencomp_shell_45.inp` with only the reference configuration imported; S3R, S4R, and S4 elements; fiber orientation 45°.

xs_s_dmg_hsntencomp_shell_45_y_y.inp	Static continuation of `xs_x_dmg_hsntencomp_shell_45.inp` with both the reference configuration and the state imported; S3R, S4R, and S4 elements; fiber orientation 45°.
xs_s_dmg_hsntencomp_cshell_0_n_n.inp	Static continuation of `xs_x_dmg_hsntencomp_cshell_0.inp` without importing the reference configuration or the state; SC6R and SC8R elements; fiber orientation 0°.
xs_s_dmg_hsntencomp_cshell_0_n_y.inp	Static continuation of `xs_x_dmg_hsntencomp_cshell_0.inp` with only the state imported; SC6R and SC8R elements; fiber orientation 0°.
xs_s_dmg_hsntencomp_cshell_0_y_n.inp	Static continuation of `xs_x_dmg_hsntencomp_cshell_0.inp` with only the reference configuration imported; SC6R and SC8R elements; fiber orientation 0°.
xs_s_dmg_hsntencomp_cshell_0_y_y.inp	Static continuation of `xs_x_dmg_hsntencomp_cshell_0.inp` with both the reference configuration and the state imported; SC6R and SC8R elements; fiber orientation 0°.
xs_s_dmg_hsntencomp_cshell_45_n_n.inp	Static continuation of `xs_x_dmg_hsntencomp_cshell_45.inp` without importing the reference configuration or the state; SC6R and SC8R elements; fiber orientation 45°.
xs_s_dmg_hsntencomp_cshell_45_n_y.inp	Static continuation of `xs_x_dmg_hsntencomp_cshell_45.inp` with only the state imported; SC6R and SC8R elements; fiber orientation 45°.
xs_s_dmg_hsntencomp_cshell_45_y_n.inp	Static continuation of `xs_x_dmg_hsntencomp_cshell_45.inp` with only the reference configuration imported; SC6R and SC8R elements; fiber orientation 45°.
xs_s_dmg_hsntencomp_cshell_45_y_y.inp	Static continuation of `xs_x_dmg_hsntencomp_cshell_45.inp` with both the reference configuration and the state imported; SC6R and SC8R elements; fiber orientation 45°.

Import from Abaqus/Standard to Abaqus/Explicit (ply fabric model)

Elements tested

CPS3 CPS4R
M3D4R
S4R SC8R

Problem description

This category of problems tests the import capability from Abaqus/Standard to Abaqus/Explicit with the ply fabric damage model. All tests subject the elements to uniaxial tension and compression loading in Abaqus/Standard. The model is then imported into Abaqus/Explicit and is subjected to further uniaxial tension and compression loading. Two off-axis angles, 0° and 45°, are considered. All the tests include problems that import only the state.

Results and discussion

The import capability is validated by comparing various damage variables and energy dissipation due to damage after each import of the results; the response after import is as expected.

Input files

sx_s_dmg_plyfabtencomp_cps_0.inp	Base problem for carrying out import from Abaqus/Standard to Abaqus/Explicit; CPS3 and CPS4R elements; off-axis angle, 0°.
sx_s_dmg_plyfabtencomp_cps_45.inp	Base problem for carrying out import from Abaqus/Standard to Abaqus/Explicit; CPS3 and CPS4R elements; off-axis angle, 45°.
sx_s_dmg_plyfabtencomp_mem_0.inp	Base problem for carrying out import from Abaqus/Standard to Abaqus/Explicit; M3D4R elements; off-axis angle, 0°.
sx_s_dmg_plyfabtencomp_mem_45.inp	Base problem for carrying out import from Abaqus/Standard to Abaqus/Explicit; M3D4R elements; off-axis angle, 45°.
sx_s_dmg_plyfabtencomp_shell_0.inp	Base problem for carrying out import from Abaqus/Standard to Abaqus/Explicit; S4R elements; off-axis angle, 0°.
sx_s_dmg_plyfabtencomp_shell_45.inp	Base problem for carrying out import from Abaqus/Standard to Abaqus/Explicit; S4R elements; off-axis angle, 45°.
sx_s_dmg_plyfabtencomp_cshell_0.inp	Base problem for carrying out import from Abaqus/Standard to Abaqus/Explicit; SC8R elements; off-axis angle, 0°.
sx_s_dmg_plyfabtencomp_cps_0.inp	Base problem for carrying out import from Abaqus/Standard to Abaqus/Explicit; SC8R elements; off-axis angle, 45°.
sx_x_dmg_plyfabtencomp_cps_0_n_y.inp	Explicit dynamic continuation of `sx_s_dmg_plyfabtencomp_cps_0.inp` with only the state imported; CPS3 and CPS4R elements; off-axis angle, 0°.
sx_x_dmg_plyfabtencomp_cps_45_n_y.inp	Explicit dynamic continuation of `sx_s_dmg_plyfabtencomp_cps_45.inp` with only the state imported; CPS3 and CPS4R elements; off-axis angle, 45°.
sx_x_dmg_plyfabtencomp_mem_0_n_y.inp	Explicit dynamic continuation of `sx_s_dmg_plyfabtencomp_mem_0.inp` with only the state imported; M3D4R elements; off-axis angle, 0°.
sx_x_dmg_plyfabtencomp_mem_45_n_y.inp	Explicit dynamic continuation of `sx_s_dmg_plyfabtencomp_mem_45.inp` with only the state imported; M3D4R elements; off-axis angle, 45°.
sx_x_dmg_plyfabtencomp_shell_0_n_y.inp	Explicit dynamic continuation of `sx_s_dmg_plyfabtencomp_shell_0.inp` with only the state imported; S4R elements; off-axis angle, 0°.
sx_x_dmg_plyfabtencomp_shell_45_n_y.inp	Explicit dynamic continuation of `sx_s_dmg_plyfabtencomp_shell_45.inp` with only the state imported; S4R elements; off-axis angle, 45°.
sx_x_dmg_plyfabtencomp_cshell_0_n_y.inp	Explicit dynamic continuation of `sx_s_dmg_plyfabtencomp_cshell_0.inp` with only the state imported; SC8R elements; off-axis angle, 0°.
sx_x_dmg_plyfabtencomp_cshell_45_n_y.inp	Explicit dynamic continuation of `sx_s_dmg_plyfabtencomp_cshell_45.inp` with only the state imported; SC8R elements; off-axis angle, 45°.

Import from Abaqus/Explicit to Abaqus/Standard (ply fabric model)

Elements tested

CPS3 CPS4R
M3D4R

S4R SC8R

Problem description

This category of problems tests the import capability from Abaqus/Explicit to Abaqus/Standard with the ply fabric damage model. All tests subject the elements to uniaxial tension and compression loading in Abaqus/Explicit. The model is then imported into Abaqus/Standard and is subjected to further uniaxial tension and compression loading. Two off-axis angles, 0° and 45°, are considered. All the tests include problems that import only the state.

Results and discussion

The import capability is validated by comparing various damage variables and energy dissipation due to damage after each import of the results; the response after import is as expected.

Input files

xs_x_dmg_plyfabtencomp_cps_0.inp	Base problem for carrying out import from Abaqus/Explicit to Abaqus/Standard; CPS3 and CPS4R elements; off-axis angle, 0°.
xs_x_dmg_plyfabtencomp_cps_45.inp	Base problem for carrying out import from Abaqus/Explicit to Abaqus/Standard; CPS3 and CPS4R elements; off-axis angle, 45°.
xs_x_dmg_plyfabtencomp_mem_0.inp	Base problem for carrying out import from Abaqus/Explicit to Abaqus/Standard; M3D4R elements; off-axis angle, 0°.
xs_x_dmg_plyfabtencomp_mem_45.inp	Base problem for carrying out import from Abaqus/Explicit to Abaqus/Standard; M3D4R elements; off-axis angle, 45°.
xs_x_dmg_plyfabtencomp_shell_0.inp	Base problem for carrying out import from Abaqus/Explicit to Abaqus/Standard; S4R elements; off-axis angle, 0°.
xs_x_dmg_plyfabtencomp_shell_45.inp	Base problem for carrying out import from Abaqus/Explicit to Abaqus/Standard; S4R elements; off-axis angle, 45°.
xs_x_dmg_plyfabtencomp_cshell_0.inp	Base problem for carrying out import from Abaqus/Explicit to Abaqus/Standard; SC8R elements; off-axis angle, 0°.
xs_x_dmg_plyfabtencomp_cshell_45.inp	Base problem for carrying out import from Abaqus/Explicit to Abaqus/Standard; SC8R elements; off-axis angle, 45°.
xs_s_dmg_plyfabtencomp_cps_0_n_y.inp	Static continuation of `xs_x_dmg_plyfabtencomp_cps_0.inp` with only the state imported; CPS3 and CPS4R elements; off-axis angle, 0°.
xs_s_dmg_plyfabtencomp_cps_45_n_y.inp	Static continuation of `xs_x_dmg_plyfabtencomp_cps_45.inp` with only the state imported; CPS3 and CPS4R elements; off-axis angle, 45°.
xs_s_dmg_plyfabtencomp_mem_0_n_y.inp	Static continuation of `xs_x_dmg_plyfabtencomp_mem_0.inp` with only the state imported; M3D4R elements; off-axis angle, 0°.
xs_s_dmg_plyfabtencomp_mem_45_n_y.inp	Static continuation of `xs_x_dmg_plyfabtencomp_mem_45.inp` with only the state imported; M3D4R elements; off-axis angle, 45°.
xs_s_dmg_plyfabtencomp_shell_0_n_y.inp	Static continuation of `xs_x_dmg_plyfabtencomp_shell_0.inp` with only the state imported; S4R elements; off-axis angle, 0°.
xs_s_dmg_plyfabtencomp_shell_45_n_y.inp	Static continuation of `xs_x_dmg_plyfabtencomp_shell_45.inp` with only the state imported; S4R elements; off-axis angle, 45°.

xs_s_dmg_plyfabtencomp_cshell_0_n_y.inp	Static continuation of `xs_x_dmg_plyfabtencomp_cshell_0.inp` with only the state imported; SC8R elements; off-axis angle, 0°.
xs_s_dmg_plyfabtencomp_cshell_45_n_y.inp	Static continuation of `xs_x_dmg_plyfabtencomp_cshell_45.inp` with only the state imported; SC8R elements; off-axis angle, 45°.

Element deletion

Elements tested

CPS4 CPS4R
M3D4
S4 S4R

Features tested

The default and nondefault degradation behaviors are tested. By default, in Abaqus/Standard elements are deleted if the damage variable for each failure mode and at each material point reaches the default maximum degradation value, $d_{max} = 1.0$. By contrast, the default behavior in Abaqus/Explicit is to delete an element when the damage variables associated with either of the fiber failure modes (tensile or compressive) reaches d_{max} at all the section points at any one integration location of an element. As an exception, a shell element (for example, S4 or S4R) is deleted in Abaqus/Explicit when all the active section points (that is, section points where the fiber failure has not been met) at any one integration location share the same through-the-thickness (z-) location. For more details of element deletion driven by material failure, see *Material Failure and Element Deletion*. You can control whether or not element deletion is activated, and you can specify the value of the damage variable at or above which a material point is assumed to be completely damaged.

Problem description

Each model consists of nine elements. A linear elastic material is assigned to all the elements except one, for which a fiber reinforced damage model is used. The specimen is subjected to biaxial extension, which is followed by biaxial compression. For each of the elements three different cases are tested:

- default behavior ($d_{max} = 1.0$, and elements are deleted if the deletion criteria are satisfied);
- default value of maximum degradation ($d_{max} = 1.0$), and the elements remain active even if the deletion criteria are satisfied; and
- the maximum degradation d_{max} is specified (0.99 for Abaqus/Standard tests; 0.975 for Abaqus/Explicit tests), and the elements remain active even if the deletion criteria are satisfied.

Results and discussion

In Abaqus/Standard simulations, the first step (biaxial extension) causes the fiber tensile and matrix tensile modes to be completely damaged. In the subsequent biaxial compression step, the remaining two failure modes (fiber and matrix compressive modes) are completely damaged as well. The evolutions of damage variables stop when the value of d_{max} is reached. Once the maximum degradation is reached at all material points for all failure modes, the elements are deleted when deletion is requested and remain active when element deletion is not requested. In Abaqus/Explicit simulations, the criterion for element deletion is met during the first step as the fibers fail in tensile mode. The element is deleted or remains active depending on whether or not element deletion is activated.

Input files

Abaqus/Standard input files

damage_elemdelete_cps4.inp	CPS4 elements are tested with default behavior (**SECTION CONTROLS*, ELEMENT DELETION=YES, $d_{max} = 1.0$).
damage_elemnodelete_cps4.inp	CPS4 elements are tested with nondefault behavior (**SECTION CONTROLS*, ELEMENT DELETION=NO, $d_{max} = 1.0$).
damage_elemnodelete099_cps4.inp	CPS4 elements are tested with nondefault behavior (**SECTION CONTROLS*, ELEMENT DELETION=NO, MAX DEGRADATION= 0.99).
damage_elemdelete_m3d4.inp	M3D4 elements are tested with default behavior (**SECTION CONTROLS*, ELEMENT DELETION=YES, $d_{max} = 1.0$).
damage_elemnodelete_m3d4.inp	M3D4 elements are tested with nondefault behavior (**SECTION CONTROLS*, ELEMENT DELETION=NO, $d_{max} = 1.0$).
damage_elemnodelete099_m3d4.inp	M3D4 elements are tested with nondefault behavior (**SECTION CONTROLS*, ELEMENT DELETION=NO, MAX DEGRADATION= 0.99).
damage_elemdelete_s4.inp	S4 elements are tested with default behavior (**SECTION CONTROLS*, ELEMENT DELETION=YES, $d_{max} = 1.0$).
damage_elemnodelete_s4.inp	S4 elements are tested with nondefault behavior (**SECTION CONTROLS*, ELEMENT DELETION=NO, $d_{max} = 1.0$).
damage_elemnodelete099_s4.inp	S4 elements are tested with nondefault behavior (**SECTION CONTROLS*, ELEMENT DELETION=NO, MAX DEGRADATION= 0.99).

Abaqus/Explicit input files

x_damage_elemdelete_cps4r.inp	CPS4R elements are tested with default behavior (**SECTION CONTROLS*, ELEMENT DELETION=YES, $d_{max} = 1.0$).
x_damage_elemnodelete_cps4r.inp	CPS4R elements are tested with nondefault behavior (**SECTION CONTROLS*, ELEMENT DELETION=NO, $d_{max} = 1.0$).
x_damage_elemnodelete0975_ cps4r.inp	CPS4R elements are tested with nondefault behavior (**SECTION CONTROLS*, ELEMENT DELETION=NO, MAX DEGRADATION= 0.975).
x_damage_elemdelete_s4r.inp	S4R elements are tested with default behavior (**SECTION CONTROLS*, ELEMENT DELETION=YES, $d_{max} = 1.0$).
x_damage_elemnodelete_s4r.inp	S4R elements are tested with nondefault behavior (**SECTION CONTROLS*, ELEMENT DELETION=NO, $d_{max} = 1.0$).
x_damage_elemnodelete0975_s4r.inp	S4R elements are tested with nondefault behavior (**SECTION CONTROLS*, ELEMENT DELETION=NO, MAX DEGRADATION= 0.975).
x_damage_elemdelete_s4.inp	S4 elements are tested with default behavior (**SECTION CONTROLS*, ELEMENT DELETION=YES, $d_{max} = 1.0$).
x_damage_elemnodelete_s4.inp	S4 elements are tested with nondefault behavior (**SECTION CONTROLS*, ELEMENT DELETION=NO, $d_{max} = 1.0$).

x_damage_elemnodelete0975_s4.inp	S4 elements are tested with nondefault behavior (**SECTION CONTROLS*, ELEMENT DELETION=NO, MAX DEGRADATION= 0.975).

Procedures

Elements tested

CPS4 CPS4R

Features tested

The Hashin damage initiation criteria with energy-based evolution law are tested with different types of procedures in Abaqus/Standard.

Problem description

This verification test consists of small models (up to nine elements) that are used with various procedure types in Abaqus/Standard. Element removal and reactivation using model change are tested by removing the element, reactivating it in the subsequent step, and verifying that all the state variables are reset correctly. The dynamic and Riks analyses are tested by comparing the numerical results with the analytical results. Finally, the linear perturbation procedures are tested by performing a general step in which the material properties are degraded before the perturbation step and then comparing the results with those obtained using a material without damage with appropriately modified parameters.

Results and discussion

The results agree well with exact analytical results or numerical results obtained using undamaged material.

Input files

damage_riks.inp	Riks analysis.
damage_modelchange.inp	Model change.
damage_freq.inp	Frequency extraction analysis.
damage_freq_undamaged.inp	Frequency extraction analysis (model without damage).
damage_dyn.inp	Dynamic analysis.
damage_ssd.inp	Steady-state dynamics.
damage_ssd_undamaged.inp	Steady-state dynamics (model without damage).

Generalized damage and failure for anisotropic materials

Products: Abaqus/Standard Abaqus/Explicit

References:

- *About Generalized Stress- and Strain-Based Damage and Failure for Anisotropic Materials*

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D4 S4R

Problem description

This verification model is a long plate with a circular hole at the center subjected to uniaxial tension, as shown in *Figure 3*. The TSAIWU and MAX STRESS damage initiation criterion are tested for this model with a number of different element types. The off-axis angle is set to −45° The material is linear elastic, and the elastic constants and the failure strength are the same as those used in *LaRC05 criterion*.

Results and discussion

The damage initiation criterion output variables TSAIWUCRT, MSTRESSCRT, TSAIWUFI, and MSTRESSFI are reviewed. The numerical results match the results obtained by hand calculations, based on the formula presented in *Generalized Damage Initiation Criteria for Anisotropic Materials*.

Plane stress orthotropic failure measures are also specified in the material when shell elements are used. In addition, stress-based theories are used. The failure index output variables MSTRS and TSAIW are compared with the damage initiation failure index output variables TSAIWUFI and MSTRESSFI, as shown in *Figure 1* and *Figure 2*. The results match as expected. The match is observed only because the loading is monotonic. Otherwise, differences are expected because the plane stress failure measures are evaluated only at the output frame, while the damage initiation failure indices are the maximum values over the entire loading history.

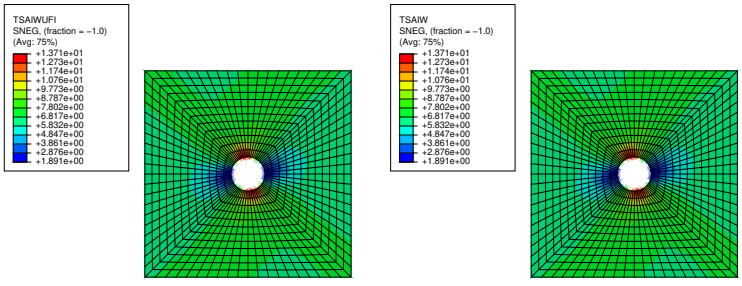


Figure 1: TSAIWUFI and TSAIW results for uniaxial tension with S4R elements.

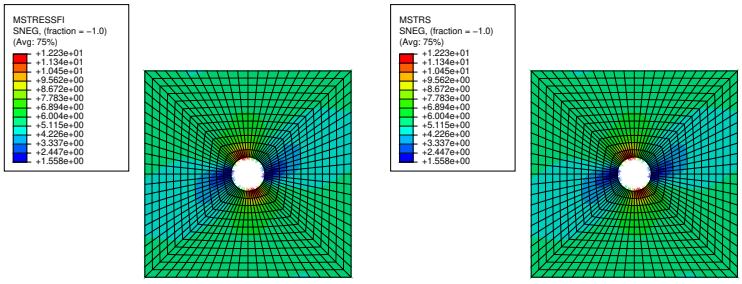


Figure 2: MSTRESSFI and MSTRS results for uniaxial tension with S4R elements.

Input files

Abaqus/Standard input files

s_aniso_dmgcrt_c3d4.inp	C3D4 elements subjected to uniaxial compression; off-axis angle, −45°.
s_aniso_dmgcrt_s4r.inp	S4R elements subjected to uniaxial compression; off-axis angle, −45°.

Abaqus/Explicit input files

x_aniso_dmgcrt_c3d4.inp	C3D4 elements subjected to uniaxial compression; off-axis angle, −45°.
x_aniso_dmgcrt_s4r.inp	S4R elements subjected to uniaxial compression; off-axis angle, −45°.

Creep

In this section:

- *Mises creep*
- *Hill creep*
- *Mises creep and plasticity*
- *Drucker-Prager creep and plasticity*
- *Cap creep and plasticity*
- *Additional verification problems for the coupled creep and plasticity capability*

Mises creep

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D8 CPS4 T3D2

Problem description

Material:

Elasticity

Young's modulus, E = 20.0E6

Poisson's ratio, ν = 0.3

Creep

Time-hardening/strain-hardening power law

- A = 2.5E−27
- n = 5.0
- m = −0.2

Hyperbolic-sine law

- A = 2.5E−27
- B = 4.4E−4
- n = 5.0
- ΔH = 0.0
- R = 8.314

Anand law

- a = 1 .6
- A = 2.5E-14
- A_0 = 1.0E4
- A_1 = -600.0
- A_2 = 0.0
- A_3 = 0.0
- A_4 = 0.0

- S_1 = 10000.0
- S_2 = -500.0
- S_3 = 0.0
- $\hat{s}$ = 1.0E-4
- Q/R = 5.0
- m = 0.5
- n = 1.0
- ξ = 1.0
- θ^Z = 0.0

Darveaux law

- B = 1.0E6
- C_{ss} = 2.25E-12
- n = 2.0
- Q/R = 5.0
- ϵ_T = 1.0E-5
- α = 2.0E-4
- θ^Z = 0.0

Double power law

- A_1 = 1.5E-10
- A_2 = 2.5E-11
- B_1 = 1.0
- B_2 = 4.0
- C_1 = 2.0
- C_2 = 5.0
- σ_0 = 10000.0
- θ^Z = 0.0

(The units are not important.)

Results and discussion

The tests in this section are set up as cases of homogeneous deformation of a single element. Consequently, the results are identical for all integration points within the element. The elements have unit dimensions except in the loading direction in which they have a length of 10. The constitutive path is integrated with the quasi-static procedure using automatic incrementation. Therefore, the number of increments varies from test to test. The results are reported at a convenient increment near the halfway point of the response and at the end of the test.

Input files

mcrtmo3qcr.inp	Time-hardening power law, uniaxial tension creep, C3D8 elements.
mcrsto3qcr.inp	Strain-hardening power law, uniaxial tension creep, C3D8 elements.
mcrhyo3qcr.inp	Hyperbolic-sine law, uniaxial tension creep, C3D8 elements.
mcrtmo3rre.inp	Time-hardening power law, uniaxial tension relaxation, C3D8 elements.
mcrsto3rre.inp	Strain-hardening power law, uniaxial tension relaxation, C3D8 elements.
mcrado3rre.inp	Anand law, uniaxial tension relaxation, C3D8 elements.
mcrdpo3rre.inp	Double power law, uniaxial tension relaxation, C3D8 elements.
mcrdxo3rre.inp	Darveaux law, uniaxial tension relaxation, C3D8 elements.
mcrtmo2qcr.inp	Time-hardening power law, uniaxial tension creep, linear perturbation with **LOAD CASE*, CPS4 elements.
mcrsto2qcr.inp	Strain-hardening power law, uniaxial tension creep, CPS4 elements.
mcrtmo2rre.inp	Time-hardening power law, uniaxial tension relaxation, CPS4 elements.
mcrsto2rre.inp	Strain-hardening power law, uniaxial tension relaxation, CPS4 elements.
mcrtmo1qcr.inp	Time-hardening power law, uniaxial tension creep, T3D2 elements.
mcrsto1qcr.inp	Strain-hardening power law, uniaxial tension creep, T3D2 elements.
mcrtmo1rre.inp	Time-hardening power law, uniaxial tension relaxation, T3D2 elements.
mcrsto1rre.inp	Strain-hardening power law, uniaxial tension relaxation, T3D2 elements.
mcrtmo3vlp.inp	Time-hardening power law, uniaxial tension creep, linear perturbation with **LOAD CASE*, C3D8 elements.

Hill creep

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D8

Problem description

Material:

Elasticity

Young's modulus, $E = 20.0E6$

Poisson's ratio, $\nu = 0.3$

Creep

$A = 2.5E{-}27$

$n = 5.0$

$m = -0.2$

Anisotropic creep ratios: 1.5, 1.2, 1.0, 1.0, 1.0, 1.0

(The units are not important.)

Results and discussion

The constitutive path is integrated with the quasi-static procedure using automatic incrementation. Therefore, the number of increments varies from test to test.

Input files

mcptmo3nt1.inp	Time-hardening power law, uniaxial tension creep in direction 1, C3D8 elements.
mcptmo3ot2.inp	Time-hardening power law, uniaxial tension creep in direction 2, C3D8 elements.
mcptmo3pt3.inp	Time-hardening power law, uniaxial tension creep in direction 3, C3D8 elements.
mcpsto3nt1.inp	Strain-hardening power law, uniaxial tension creep in direction 1, C3D8 elements.
mcpsto3ot2.inp	Strain-hardening power law, uniaxial tension creep in direction 2, C3D8 elements.
mcpsto3pt3.inp	Strain-hardening power law, uniaxial tension creep in direction 3, C3D8 elements.

mcptmo3vlp.inp	Time-hardening power law, uniaxial tension creep in direction 1, linear perturbation with **LOAD CASE*, C3D8 elements.

Mises creep and plasticity

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

B32 C3D8 C3D8R CPS4 S4 S4R T3D2

Problem description

Material:

Elasticity

Young's modulus, E = 20.0E6

Poisson's ratio, ν = 0.3

Plasticity

Hardening curve:

Yield stress	Plastic strain
10.0E3	0.00
50.0E3	0.02

Creep

A = 1.0E−24

n = 5.0

m = −0.2

Swelling

Volumetric swelling rate = 2.0E−6

(The units are not important.)

Results and discussion

The tests in this section verify the coupled Mises creep and plasticity model for problems involving uniaxial tension, shear, bending, and torsion. The test cases consider stress spaces with 1, 2, or 3 direct components. Both time and strain creep laws, as well as volumetric swelling, are considered with the constitutive path integrated by the quasi-static procedure using automatic incrementation. Explicit and implicit time integration are employed, with automatic switching to the implicit scheme once a material point goes plastic. The solution's accuracy is verified by comparing it to test cases employing extremely fine time integration.

Input files

mmctmo1hut.inp	Time-hardening power law, uniaxial tension creep, T3D2 elements.
mmctmo2hut.inp	Time-hardening power law, uniaxial tension creep, CPS4 elements.
mmctmo2euc.inp	Time-hardening power law, uniaxial compression creep, linear perturbation with **LOAD CASE*, S4R elements.
mmctmo2euce.inp	Time-hardening power law, uniaxial compression creep, S4 elements.
mmctmo3hut.inp	Time-hardening power law, uniaxial tension creep, C3D8R elements.
mmcsto3hut.inp	Strain-hardening power law, uniaxial tension creep, C3D8R elements.
mkcsto3hut.inp	Strain-hardening power law, uniaxial tension creep, Hardening=Kinematic, C3D8R elements.
mmcsto2gsh.inp	Strain-hardening power law, shear creep, CPS4 elements.
mmcsto3gsh.inp	Strain-hardening power law, shear creep, C3D8 elements.
mswooo1ahc.inp	Volumetric swelling, T3D2 elements.
mswooo2ahc.inp	Volumetric swelling, CPS4 elements.
mswooo3ahc.inp	Volumetric swelling, C3D8 elements.
mmcsto1xmx.inp	Time-hardening power law, creep law, combined torsion and bending, B32 elements.

Drucker-Prager creep and plasticity

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D8 C3D8R

Problem description

Material:

Elasticity

Young's modulus, $E = 300.0E3$

Poisson's ratio, $\nu = 0.3$

Plasticity

Angle of friction, $\beta = 40.0$

Dilation angle, $\psi = 40.0$

Third invariant ratio, $K = 1.0$

Hardening curve:

Yield stress	Plastic strain
6.0E3	0.00
9.0E3	0.02
11.0E3	0.063333
12.0E3	0.11
12.0E3	1.0

Creep

For the time and strain creep laws:

- $A = 0.5E{-}7$
- $n = 1.1$
- $m = -0.2$

The Singh-Mitchell creep law parameters are varied. For example:

- $A = 0.002$
- $\alpha = 1.0E{-}6$

- $m = -1.0$
- $\boldsymbol{t1} = 1.0$

(The units are not important.)

Results and discussion

The tests in this section verify the coupled Drucker-Prager creep and plasticity model. The tests are set up as cases of homogeneous deformation of a single solid element of unit dimension subjected to uniaxial tension and compression, shear, and hydrostatic tension. The Singh-Mitchell, time, and strain hardening creep laws are considered with the constitutive path integrated by the quasi-static procedure. Explicit and implicit time integration are employed, with automatic switching to the implicit scheme once a material point goes plastic.

Input files

mdcsmo3euc.inp	Singh-Mitchell type law, uniaxial compression, C3D8 elements.
mdcsmo3hut.inp	Singh-Mitchell type law, uniaxial tension, C3D8 elements.
mdcsmo3gsh.inp	Singh-Mitchell type law, shear, C3D8R elements.
mdcsmo3jht.inp	Singh-Mitchell type law, hydrostatic tension, C3D8R elements.
mdcsmt3euc.inp	Singh-Mitchell type law, uniaxial compression with temperature dependence, C3D8 elements.
mdctmo3hut.inp	Time-hardening power law, uniaxial tension, C3D8 elements.
mdcsto3hut.inp	Strain-hardening power law, uniaxial tension, perturbation step with **LOAD CASE*, C3D8 elements.
mdcuco3hut.inp	User-specified creep law, uniaxial tension, C3D8 elements.
mdcuco3hut.f	User subroutine *CREEP* used in mdcuco3hut.inp.

Cap creep and plasticity

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D8 C3D8R

Problem description

Material:

Elasticity

Young's modulus, $E = 300.0E4$

Poisson's ratio, $\nu = 0.3$

Cap plasticity

Material cohesion, $d = 2.0E4$

Material angle of friction, $\beta = 40.0$

Cap eccentricity, $R = 0.3$

Initial cap yield surface, $\epsilon_{vol}^{pl}(0) = 0.5$

Transition surface radius, $\alpha = 0.0$

Third invariant ratio, $K = 1.0$

Hardening curve:

Hydrostatic pressure yield stress	Volumetric plastic strain
6.01E3	0.0
6.04E3	0.4
1.432E4	0.5
3.5E4	0.7
8.7E4	1.0

Creep (for both cohesion and consolidation)

Time-hardening power law:

- $A = 1.0E{-}24$
- $n = 5$
- $m = 0.0$

Strain-hardening power law:

- $A = 7.0\text{E}{-}26$
- $n = 5$
- $m = 0.0$

Singh-Mitchell type law:

- $A = 0.002$
- $\alpha = 1.6\text{E}{-}4$
- $m = 0.0$
- $t1 = 1.0$

User-specified creep law:

The user subroutine for the time creep law specified earlier is implemented.

(The units are not important.)

Results and discussion

The tests in this section verify the cap creep and plasticity model. The tests are set up as cases of homogeneous deformation of a single solid element of unit dimension. To validate the model, the element is subjected to various stress paths including uniaxial tension and compression, shear, hydrostatic tension and compression, and triaxial compression. The Singh-Mitchell creep law, the time and strain hardening creep laws, and a user-defined creep model are considered with the constitutive path integrated by the quasi-static procedure. Explicit and implicit time integration are employed, with automatic switching to the implicit scheme once a material point goes plastic.

Input files

mccsmo3ahc.inp	Singh-Mitchell type law, hydrostatic compression, C3D8R elements.
mccsmo3euc.inp	Singh-Mitchell type law, uniaxial compression, C3D8 elements.
mccsto3hut.inp	Strain-hardening power law, uniaxial tension, C3D8 elements.
mccsto3gsh.inp	Strain-hardening power law, shear, C3D8R elements.
mcctmo3aht.inp	Time-hardening power law, hydrostatic tension, C3D8R elements.
mcctmo3ctc.inp	Time-hardening power law, triaxial compression, C3D8R elements.
mccuco3ctc.inp	User-specified creep law, triaxial compression, C3D8R elements.
mccuco3ctc.f	User subroutine *CREEP* used in mccuco3ctc.inp.

Additional verification problems for the coupled creep and plasticity capability

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

CAX8R CINPE5R CPE4 CPE8R

Problem description

Additional verification problems were obtained by adding creep to the plasticity model of *Limit load calculations with granular materials* and *Finite deformation of an elastic-plastic granular material*. For these cases a small creep strain rate was selected to verify the plasticity component of the coupled creep and plasticity models. Thus, the results should be comparable to the equivalent problem without creep, although they are separate Abaqus material models. These verification problems test both the Drucker-Prager creep and the Drucker-Prager/Cap creep models.

Further verification problems for Mises creep and plasticity were obtained by adding plasticity to the problems described in *Creep of a thick cylinder under internal pressure* and *Ct-integral evaluation*. For the example described in *Creep of a thick cylinder under internal pressure*, the initial application of the pressure plastifies the cylinder during the first step of the analysis; and the creep response is then developed in the second step. For the example described in *Ct-integral evaluation*, the plastic deformation is very small and localized. Plastification occurs only during the preloading static step. As a result, the C_t-integrals calculated by Abaqus in the early stages of the quasi-static step are expected to differ somewhat from the ones calculated in the creep-only case and are not path independent. Later on, when larger scale creep dominates the stress fields, the C_t-integrals calculated should converge toward the same values as obtained in the creep-only case and become path independent.

Results and discussion

The results obtained show good agreement with the corresponding example problems. The addition of creep in the first two problems has little effect on the plastic results, and the addition of plasticity in the second two problems has little effect on the creep results.

Input files

granmatlimitload1.inp	Verification input file for the problem described in *Limit load calculations with granular materials*.
granmatlimitload2.inp	Verification input file for the problem described in *Limit load calculations with granular materials*.
granmatfinitedef1.inp	Verification input file for the problem described in *Finite deformation of an elastic-plastic granular material*.
thickcylcreep1.inp	Verification input file for the problem described in *Creep of a thick cylinder under internal pressure*.
ctintegral1.inp	Verification input file for the problem described in *Ct-integral evaluation*.

Concrete smeared cracking

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D8 C3D8H C3D8R CPS4 CPS4R T3D2

Problem description

Material:

Elasticity

Young's modulus, E = 4.65E6

Poisson's ratio, ν = 0.18

Plasticity

Biaxial/uniaxial compression stress ratio = 1.18

Uniaxial tension/compression stress ratio = 0.1

Biaxial/uniaxial compression plastic strain ratio = 1.25

Tensile cracking stress/compression stress ratio = 0.2

Hardening curve:

Yield stress	Plastic strain
1300.0	0.000000
2200.0	0.000027
3000.0	0.000100
3600.0	0.000225
4450.0	0.000550
4650.0	0.001000
4200.0	0.002000
2000.0	0.003500

Cracking tensile stress decreases linearly to reach zero at strain = 5.0E−4.

For constant fracture energy cracking cases (mcou0o[1/2/3]hut.inp) cracking tensile stress decreases linearly to reach zero at displacement = 6.0E−4.

(The units are not important.)

Results and discussion

The results agree well with exact analytical or approximate solutions.

Input files

mcoooo3euc.inp	Uniaxial compression, C3D8 elements.
mcoooo2euc.inp	Uniaxial compression, CPS4 elements.
mcoooo3fbc.inp	Biaxial compression, C3D8 elements.
mcoooo2fbc.inp	Biaxial compression, CPS4 elements.
mcoooo3gsh.inp	Shear, C3D8 elements.
mcoooo3hut.inp	Uniaxial tension, C3D8 elements.
mcoooo2hut.inp	Uniaxial tension, CPS4 elements.
mcoooo1hut.inp	Uniaxial tension, T3D2 elements.
mcoooo3ibt.inp	Biaxial tension, C3D8H elements.
mcoooo2ibt.inp	Biaxial tension, CPS4 elements.
mcoooo3jht.inp	Hydrostatic tension, C3D8H elements.
mcoooo3kct.inp	Compression tension, C3D8 elements.
mcooot3euc.inp	Uniaxial compression, C3D8 elements, temperature dependence.
mcoood3fbc.inp	Biaxial compression, C3D8 elements, field variable dependence.
mcooob3hut.inp	Uniaxial tension, C3D8 elements, temperature and field variable dependence.
mcou0o3hut.inp	Uniaxial tension, C3D8R elements, constant fracture energy cracking.
mcou0o2hut.inp	Uniaxial tension, CPS4R elements, constant fracture energy cracking.
mcou0o1hut.inp	Uniaxial tension, T3D2 elements, constant fracture energy cracking.
mcoooo3vlp.inp	Perturbation step with **LOAD CASE*, uniaxial tension, C3D8 elements.
mcooot3vlp.inp	Linear perturbation uniaxial compression, C3D8 elements, temperature dependence.

Concrete damaged plasticity

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D8R CPE4R CPS4R T3D2

Problem description

Material:

Elasticity

Young's modulus, E = 2.648E+10

Poisson's ratio, ν = 0.167

Plasticity

Dilation angle ϕ = 15.0

Flow potential eccentricity ϵ = 0.1

Biaxial/uniaxial compression plastic strain ratio σ_{b0}/σ_{c0} = 1.16

Invariant stress ratio K_c = 0.6667

Viscosity μ = 0.0

Compression behavior:

Stress	Inelastic strain	Damage	Inelastic strain
24.019E+6	0.0000	0.0000	0.0000
29.208E+6	0.0004	0.1299	0.0004
31.709E+6	0.0008	0.2429	0.0008
32.358E+6	0.0012	0.3412	0.0012
31.768E+6	0.0016	0.4267	0.0016
30.379E+6	0.0020	0.5012	0.0020
28.507E+6	0.0024	0.5660	0.0024
21.907E+6	0.0036	0.7140	0.0036
14.897E+6	0.0050	0.8243	0.0050
2.953E+6	0.0100	0.9691	0.0100

Tension recovery factor w_t = 0.0.

Tension behavior:

Stress	Cracking strain	Damage	Cracking strain
1.780E+6	0.0000	0.0000	0.0000
1.457E+6	0.0001	0.3000	0.0001
1.113E+6	0.0003	0.5500	0.0003
0.960E+6	0.0004	0.7000	0.0004
0.800E+6	0.0005	0.8000	0.0005
0.536E+6	0.0008	0.9000	0.0008
0.359E+6	0.0010	0.9300	0.0010
0.161E+6	0.0020	0.9500	0.0020
0.073E+6	0.0030	0.9700	0.0030
0.040E+6	0.0050	0.9900	0.0050

Compression recovery factor w_c = 1.0.

Other properties

Density = 2400.0

(The units are not important.)

Results and discussion

The results agree well with exact analytical or approximate solutions.

Input files

Abaqus/Standard input files

cdpse_c3d8r.inp	Uniaxial cyclic test, C3D8R elements, TYPE=STRAIN.
cdpse_cpe4r.inp	Uniaxial cyclic test, CPE4R elements, TYPE=STRAIN.
cdpse_cps4r.inp	Uniaxial cyclic test, CPS4R elements, TYPE=STRAIN.
cdpse_t3d2.inp	Uniaxial cyclic test, T3D2 elements, TYPE=STRAIN.
cdpsu_c3d8r.inp	Uniaxial cyclic test, C3D8R elements, TYPE=DISPLACEMENT.
cdpsu_cpe4r.inp	Uniaxial cyclic test, CPE4R elements, TYPE=DISPLACEMENT.
cdpsu_cps4r.inp	Uniaxial cyclic test, CPS4R elements, TYPE=DISPLACEMENT.
cdpsu_t3d2.inp	Uniaxial cyclic test, T3D2 elements, TYPE=DISPLACEMENT.
cdpseo_c3d8r.inp	Uniaxial cyclic test, C3D8R elements, TYPE=STRAIN, with ORIENTATION and linear perturbation step.
cdpseo_cpe4r.inp	Uniaxial cyclic test, CPE4R elements, TYPE=STRAIN, with ORIENTATION and linear perturbation step.
cdpseo_cps4r.inp	Uniaxial cyclic test, CPS4R elements, TYPE=STRAIN, with ORIENTATION and linear perturbation step.
cdpseb_c3d8r.inp	Biaxial cyclic test, C3D8R elements, TYPE=STRAIN.
cdpseb_cpe4r.inp	Biaxial cyclic test, CPE4R elements, TYPE=STRAIN.
cdpseb_cps4r.inp	Biaxial cyclic test, CPS4R elements, TYPE=STRAIN.

cdpses_c3d8r.inp	Shear cyclic test, C3D8R elements, TYPE=STRAIN.
cdpses_cpe4r.inp	Shear cyclic test, CPE4R elements, TYPE=STRAIN.
cdpses_cps4r.inp	Shear cyclic test, CPS4R elements, TYPE=STRAIN.

Abaqus/Explicit input files

cdpxe_c3d8r.inp	Uniaxial cyclic test, C3D8R elements, TYPE=STRAIN.
cdpxe_cpe4r.inp	Uniaxial cyclic test, CPE4R elements, TYPE=STRAIN.
cdpxe_cps4r.inp	Uniaxial cyclic test, CPS4R elements, TYPE=STRAIN.
cdpxe_t3d2.inp	Uniaxial cyclic test, T3D2 elements, TYPE=STRAIN.
cdpxu_c3d8r.inp	Uniaxial cyclic test, C3D8R elements, TYPE=DISPLACEMENT.
cdpxu_cpe4r.inp	Uniaxial cyclic test, CPE4R elements, TYPE=DISPLACEMENT.
cdpxu_cps4r.inp	Uniaxial cyclic test, CPS4R elements, TYPE=DISPLACEMENT.
cdpxu_t3d2.inp	Uniaxial cyclic test, T3D2 elements, TYPE=DISPLACEMENT.
cdpxeo_c3d8r.inp	Uniaxial cyclic test, C3D8R elements, TYPE=STRAIN, with ORIENTATION.
cdpxeo_cpe4r.inp	Uniaxial cyclic test, CPE4R elements, TYPE=STRAIN, with ORIENTATION.
cdpxeo_cps4r.inp	Uniaxial cyclic test, CPS4R elements, TYPE=STRAIN, with ORIENTATION.
cdpxeb_c3d8r.inp	Biaxial cyclic test, C3D8R elements, TYPE=STRAIN.
cdpxeb_cpe4r.inp	Biaxial cyclic test, CPE4R elements, TYPE=STRAIN.
cdpxeb_cps4r.inp	Biaxial cyclic test, CPS4R elements, TYPE=STRAIN.
cdpxes_c3d8r.inp	Shear cyclic test, C3D8R elements, TYPE=STRAIN.
cdpxes_cpe4r.inp	Shear cyclic test, CPE4R elements, TYPE=STRAIN.
cdpxes_cps4r.inp	Shear cyclic test, CPS4R elements, TYPE=STRAIN.

Transferring results from Abaqus/Standard to Abaqus/Explicit

cdp_c3d8r_ss_s1.inp	Abaqus/Standard base analysis.
cdp_c3d8r_ss_s2.inp	Abaqus/Standard import analysis from cdp_c3d8r_ss_s1.inp.
cdp_c3d8r_sx_s.inp	Abaqus/Standard base analysis.
cdp_c3d8r_sx_x.inp	Abaqus/Explicit import analysis from cdp_c3d8r_sx_s.inp.
cdp_c3d8r_xs_s.inp	Abaqus/Standard import analysis from cdp_c3d8r_sx_x.inp.
cdpeo_cpe4r_sx_s.inp	Abaqus/Standard base analysis.
cdpeo_cpe4r_sx_x.inp	Abaqus/Explicit import analysis from cdpeo_cpe4r_sx_s.inp.
cdpeo_cpe4r_xs_s.inp	Abaqus/Standard import analysis from cdpeo_cpe4r_sx_x.inp.
cdps_c3d8r_sx_s.inp	Abaqus/Standard base analysis.
cdps_c3d8r_sx_x.inp	Abaqus/Explicit import analysis from cdps_c3d8r_sx_s.inp.
cdps_c3d8r_xs_s.inp	Abaqus/Standard import analysis from cdps_c3d8r_sx_x.inp.
cdpb_c3d8r_sx_s.inp	Abaqus/Standard base analysis.
cdpb_c3d8r_sx_x.inp	Abaqus/Explicit import analysis from cdpb_c3d8r_sx_s.inp.
cdpb_c3d8r_xs_s.inp	Abaqus/Standard import analysis from cdpb_c3d8r_sx_x.inp.
cdpu_cps4r_sx_s.inp	Abaqus/Standard base analysis.
cdpu_cps4r_sx_x.inp	Abaqus/Explicit import analysis from cdpu_cps4r_sx_s.inp.

cdpu_cps4r_xs_s.inp	Abaqus/Standard import analysis from cdpu_cps4r_sx_x.inp.
cdpe_t3d2_sx_s.inp	Abaqus/Standard base analysis.
cdpe_t3d2_sx_x.inp	Abaqus/Explicit import analysis from cdpe_t3d2_sx_s.inp.
cdpe_t3d2_xs_s.inp	Abaqus/Standard import analysis from cdpe_t3d2_sx_x.inp.

Two-layer viscoplasticity

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D8R CPS4R SAX1

Problem description

Material:

Elasticity

Young's modulus, E=20.0E6

Poisson's ratio, ν=0.3

Plasticity 1

Mises perfect plasticity with yield stress = 200.0

Plasticity 2

Mises plasticity with linear kinematic hardening

Hardening:

Yield stress	Plastic strain
200.	0.0000
220.	0.0009

Plasticity 3

Hill perfect plasticity with reference yield stress = 200.0

Anisotropic yield ratios: 1.0, 1.0, 1.0, 1.0, 1.0, 1.0

Because of the choice of the anisotropic yield ratios, the material represented by plasticity 3 is identical to the material represented by plasticity 1.

Viscous 1

LAW=TIME

- A=1.0E−6
- n=1.0

- m=0.0
- f=0.25

Viscous 2

LAW=STRAIN

- A=1.0E−6
- n=1.0
- m=0.0
- f=0.25

Viscous 3

LAW=TIME

- A=1.0E−6
- n=1.0
- m=0.0
- f=0.25

Anisotropic viscosity ratios: 1.0, 1.0, 1.0, 1.0, 1.0, 1.0

Because of the choice of the anisotropic viscosity ratios, the material represented by viscous 3 is identical to the material represented by viscous 1.

(The units are not important.)

Results and discussion

The results agree well with exact analytical or approximate solutions.

Input files

mtvtmt3rre.inp	Plasticity 1 and Viscous 1; uniaxial tension followed by relaxation; strain controlled; tests temperature dependence of viscous data; tests linear perturbation; C3D8R elements.
mtvtmt2rre.inp	Plasticity 1 and Viscous 1; uniaxial tension followed by relaxation; strain controlled; CPS4R elements.
mtvtmo2rre.inp	Plasticity 1 and Viscous 1; uniaxial tension followed by relaxation; strain controlled; SAX1 elements.
mtvtmo2rre_po.inp	**POST OUTPUT* analysis.
mtvtmt3pt3.inp	Plasticity 1 and Viscous 1; uniaxial tension; long-term static solution; C3D8R elements.
mtvsto3mcy.inp	Plasticity 2 and Viscous 2; uniaxial cyclic loading; strain controlled; C3D8R elements.
mtvvpo3mcy.inp	Plasticity 3 and Viscous 3; Uniaxial cyclic loading; strain controlled; C3D8R elements.

Brittle cracking constitutive model

Products: Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

CAX4R C3D8R CPS4R CPE4R B21

Features tested

Brittle cracking model response under loading/unloading/reloading conditions. Different combinations of active cracks are tested.

Problem description

This problem contains 21 single-element verification problems that are all run in one input file. The problem exercises the brittle cracking material model under loading/unloading/reloading conditions; all possible crack states are exercised for single and multiple crack cases.

Figure 1 shows the 21 elements used in the analysis in their original and deformed shapes. The dashed lines illustrate the original shapes. The bottom row contains CAX4R and C3D8R elements only since they are the only elements for which it is not possible to create three simultaneous cracks. The next row up contains all but B21 elements since they are the only elements for which it is not possible to create two simultaneous cracks. The top three rows contain all five element types since they refer to loading cases resulting in a single crack. The three rows are used to test the three different ways available for input of the tension softening data.

The original length of each side of the elements is $L_0 = 1$. The elements are loaded using an amplitude function that subjects them to tension, followed by unloading and loading into compression, followed by reloading in tension. This loading program is applied in one direction (rows (c), (d), and (e) in *Figure 1*), two directions (row (b)) or three directions (row (a)). This creates one, two, or three simultaneous cracks, respectively.

The material properties used are those of a typical medium strength concrete: the elastic properties are $E = 30 \times 10^9$ Pa, $\nu = 0.2$; the cracking failure stress is 3×10^6 Pa; and the mass density is 2400 kg/m^3.

Results and discussion

Figure 2 shows stress-strain in all three cracking directions for elements CAX4R and C3D8R (row (a) in *Figure 1*). For CAX4R the radial and axial loading is applied equally. For C3D8R directions 1 and 3 are loaded at the same rate, whereas direction 2 is loaded at three-quarters of that rate. The results for the two kinds of elements are identical.

Figure 3 shows stress-strain in two cracking directions for elements CAX4R, C3D8R, CPS4R, and CPE4R (row (b) in *Figure 1*). For all but the axisymmetric case, direction 2 is loaded at three-quarters of the loading rate in direction 1. In the axisymmetric case the radial and axial directions are loaded at the same rate. The results for all elements are in agreement.

Figure 4 shows stress-strain in the only cracking direction (direction 2) for elements CAX4R, C3D8R, CPS4R, CPE4R, and B21 (row (c) in *Figure 1*). The tension softening data are defined by specifying the postfailure stress-strain relationship. The results for all elements are identical.

Figure 5 shows stress-strain in the only cracking direction (direction 2) for elements CAX4R, C3D8R, CPS4R, CPE4R, and B21 (row (d) in *Figure 1*). The tension softening data are defined by specifying the postfailure stress/displacement relationship. The results for all but the axisymmetric element are identical. The axisymmetric result is slightly different because the characteristic length computed by Abaqus/Explicit is different in the axisymmetric case.

Figure 6 shows stress-strain in the only cracking direction (direction 2) for elements CAX4R, C3D8R, CPS4R, CPE4R, and B21 (row (e) in *Figure 1*). The tension softening data are defined by specifying the failure stress, σ_{tu}^{I}, and the Mode 1 fracture energy, G_f^I. The results for all but the axisymmetric element are identical. The axisymmetric result is slightly different because the characteristic length computed by Abaqus/Explicit is different in the axisymmetric case.

Input files

cracking.inp	Input data used in this analysis.

Figures

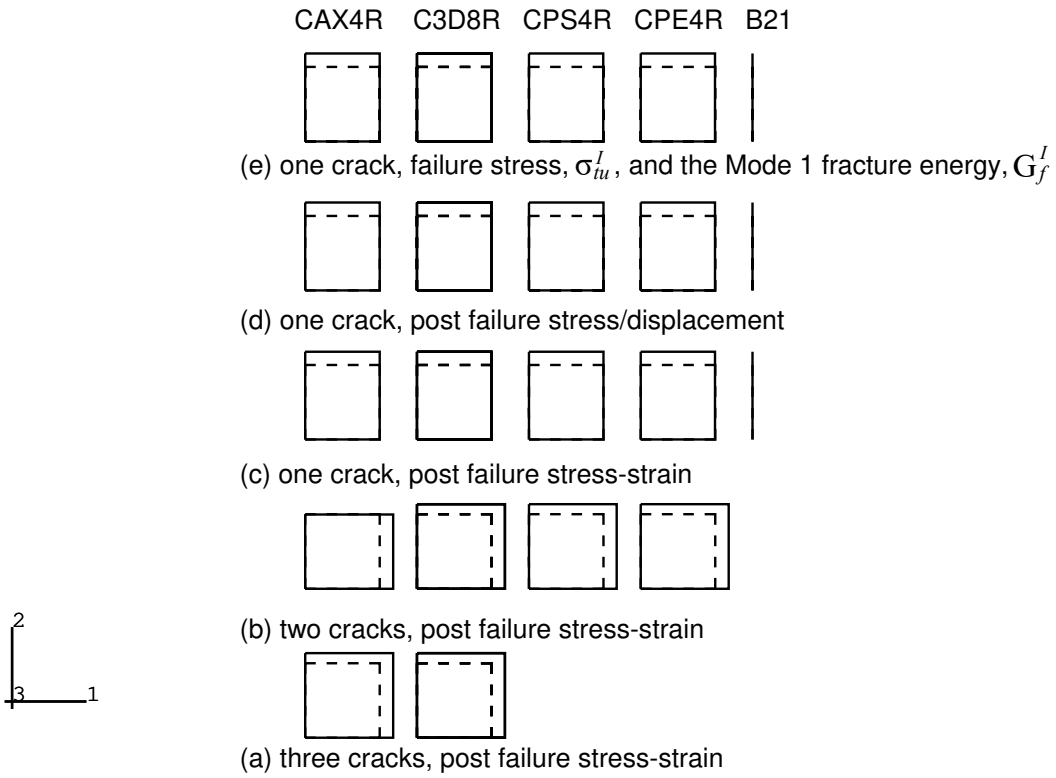


Figure 1: Deformed shapes for one-element cracking model tests.

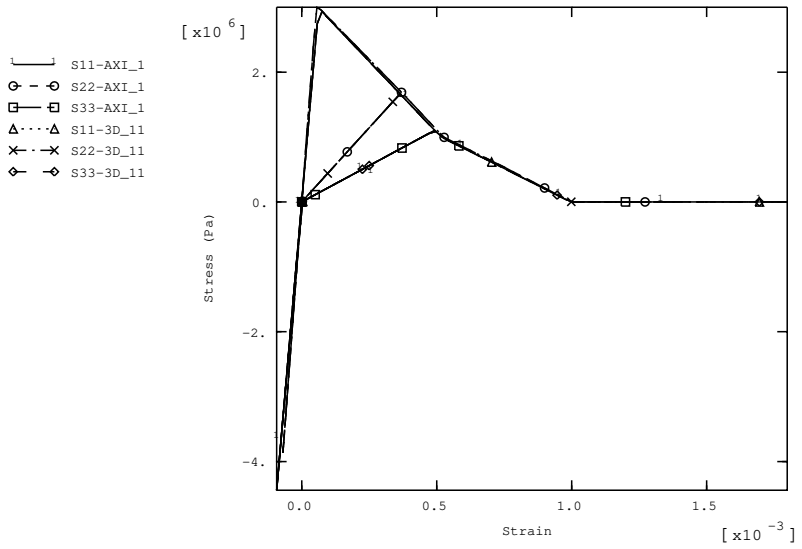


Figure 2: Stress-strain in three cracking directions: CAX4R and C3D8R elements.

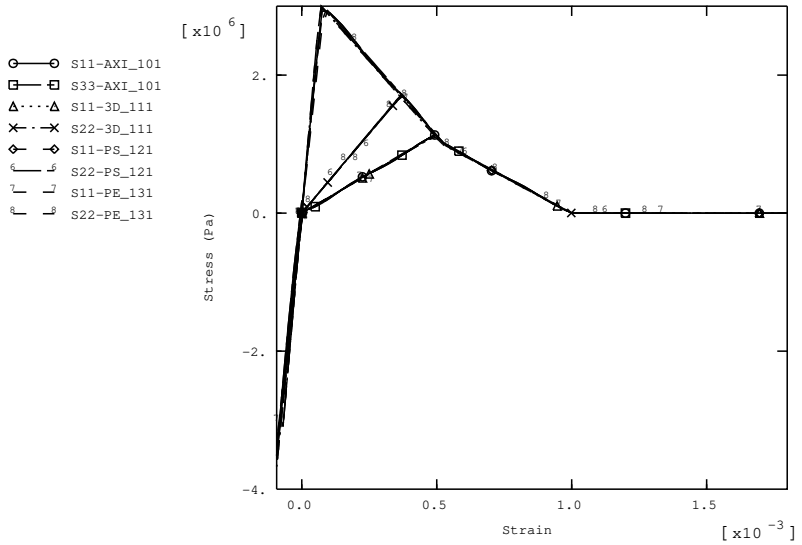


Figure 3: Stress-strain in two cracking directions: CAX4R, C3D8R, CPS4R, and CPE4R elements.

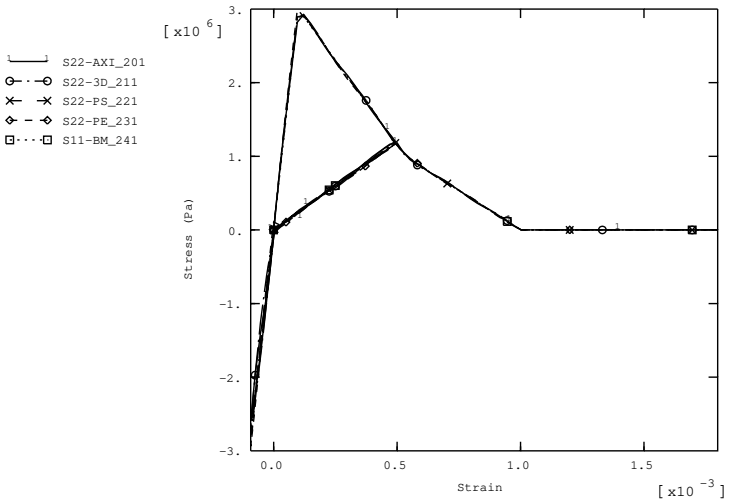


Figure 4: Stress-strain in single cracking direction (stress-strain specified): CAX4R, C3D8R, CPS4R, CPE4R, and B21 elements.

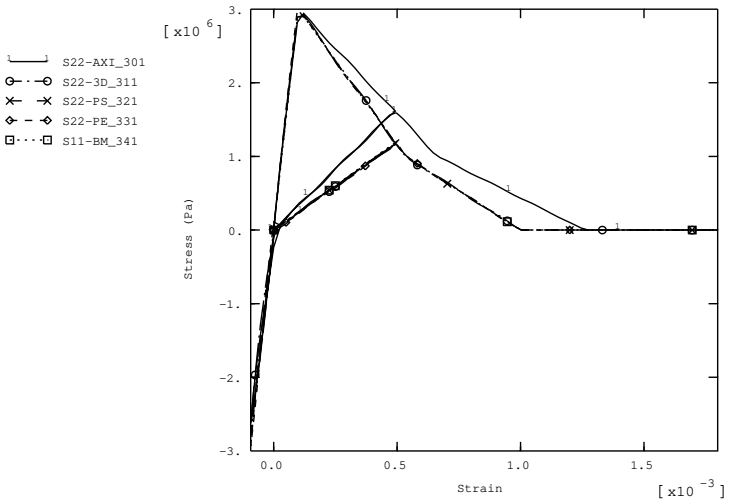


Figure 5: Stress-strain in single cracking direction (stress/displacement specified): CAX4R, C3D8R, CPS4R, CPE4R, and B21 elements.

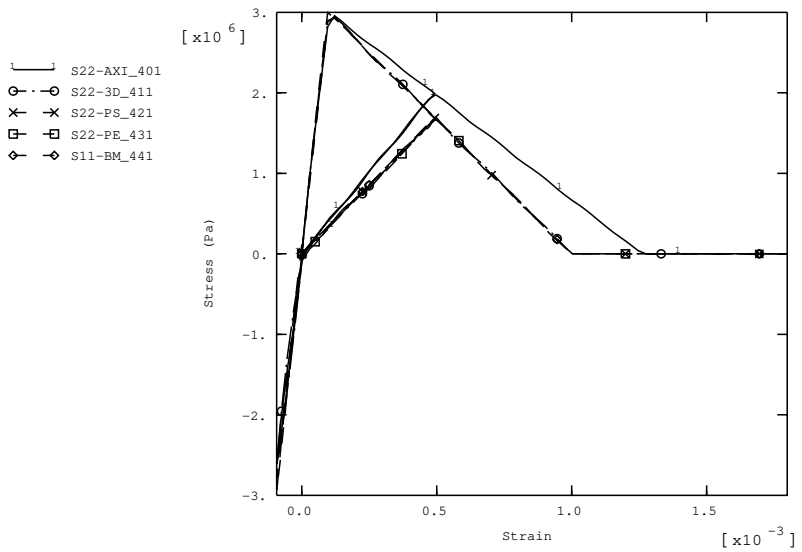


Figure 6: Stress-strain in single cracking direction (failure stress and fracture energy specified): CAX4R, C3D8R, CPS4R, CPE4R, and B21 elements.

Cracking model: tension shear test

Products: Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D8R CPS4R CPE4R

Features tested

Brittle cracking model response under simultaneous tension and shear loading: this verifies the shear retention formulation used in the model.

Problem description

This test illustrates the behavior of the brittle cracking model when subjected to simultaneous tension and shear loading. This behavior has been the subject of much discussion in the context of comparing different kinds of cracking models (fixed cracks versus rotating cracks, orthogonal cracks versus non-orthogonal cracks); see, for example, *Rots and Blaauwendraad (1989)*. It has been argued that fixed orthogonal crack models, such as the one implemented in Abaqus/Explicit, produce shear behavior that is too stiff. In this verification example we show that this is not the case because of the manner in which the shear retention behavior is formulated in Abaqus/Explicit (as described in *A cracking model for concrete and other brittle materials*).

The test carried out here was originally suggested by *Willam et al. (1987)*. It consists of loading a specimen in the horizontal direction (direction 1) until a vertical crack initiates (*Figure 1*(a)); then the specimen is loaded simultaneously in biaxial tension and shear, as shown in *Figure 1*(b).The latter part of the loading causes the principal stress directions to rotate, and the issue is whether the cracking model provides an adequate shear response (the shear stress must vanish as deformation takes place).

This test is carried out on six single elements that are all run in one input file. The original length of each side of the elements is $L_0 = 1$. *Figure 2* shows the six elements used in the analysis in their original and deformed shapes. The dashed lines illustrate the original shapes. The bottom row contains C3D8R, CPS4R, and CPE4R elements with shear retention properties defined using a power law analytical form, while the top row contains the same elements but with shear retention properties defined using a tabular form that mimics the analytical form. The purpose of testing the two groups of elements is to verify the two different options available in Abaqus/Explicit for defining shear retention.

The material properties used are those of a typical medium strength concrete: the elastic properties are $E = 30 \times 10^9$ Pa, $\nu = 0.2$; the cracking failure stress is 3×10^6 Pa; the shear retention is defined by the power law provided in Abaqus/Explicit with $p = 2$ and $e^{ck}_{max} = .001$; and the mass density is 2400 kg/m^3.

Results and discussion

Figure 3 shows horizontal stress-strain for the three different element types using the power law shear retention input definition. The results are identical for the three element types. *Figure 4* shows horizontal stress-strain for the three different element types using the tabular shear retention input definition. The results are again identical for the three element types. In addition, comparing the results for the two different shear retention input definitions,

we observe that they are identical. *Figure 5* and *Figure 6* show similar results for vertical stress-strain behavior. This horizontal and vertical stress-strain behavior obtained with the cracking model is a reflection of the input tension softening data, since the specimen cracks both in the horizontal and vertical directions.

Figure 7 shows shear stress-strain for the three different element types using the power law shear retention input definition. The results are identical for the three element types. *Figure 8* shows shear stress-strain for the three different element types using the tabular shear retention input definition. The results are again identical for the three element types. In addition, comparing the results for the two different shear retention input definitions, we observe that they are identical. We also observe that the model provides shear stress that increases to a maximum value (which depends on the shear retention properties) and then decreases to zero. This damage-like shear behavior is an important characteristic, and it has been claimed that rotating crack models provide it, while fixed crack models cannot. This test shows that the cracking model implemented in Abaqus/Explicit does produce this desired shear behavior.

Input files

cracking_ts.inp	Input data used in this analysis.

References

- Rots, J. G., and J. Blaauwendraad, “Crack Models for Concrete: Discrete or Smeared? Fixed, Multi-Directional or Rotating?,” HERON, vol. 34, no. 1, Delft University of Technology, The Netherlands, 1989.
- Willam, K., E. Pramono, and S. Sture, “Fundamental Issues of Smeared Crack Models,” Proc. SEM–RILEM International Conference on Fracture of Concrete and Rock, S.P. Shah and S.E. Swartz (Eds.), SEM, Bethel, pp. 192–207, 1987.

Figures

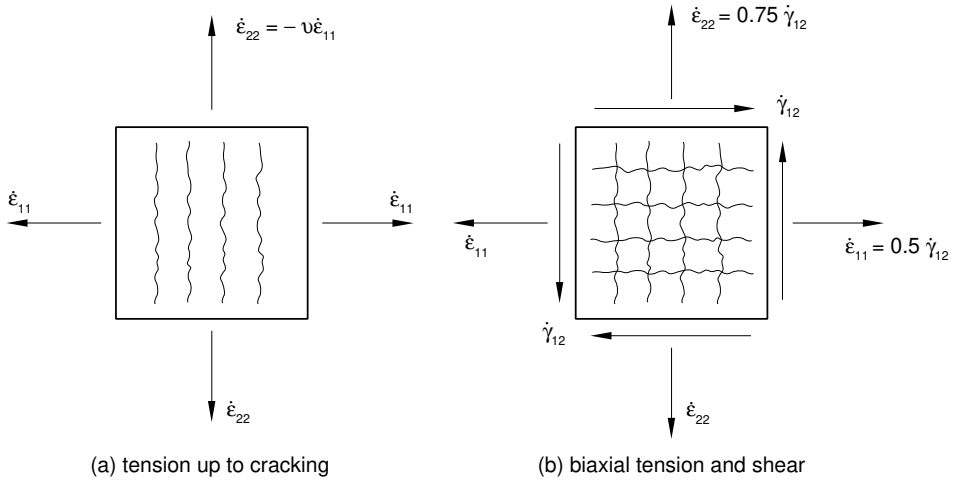


Figure 1: Tension shear test loading sequence.

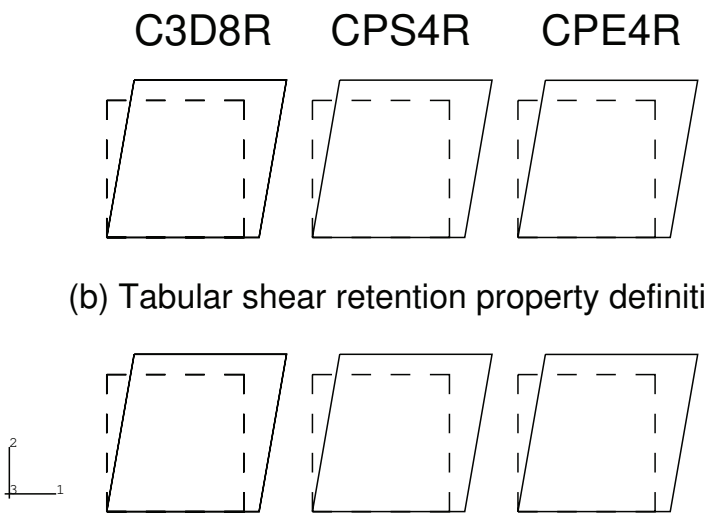


Figure 2: Deformed shapes for one-element tension shear tests.

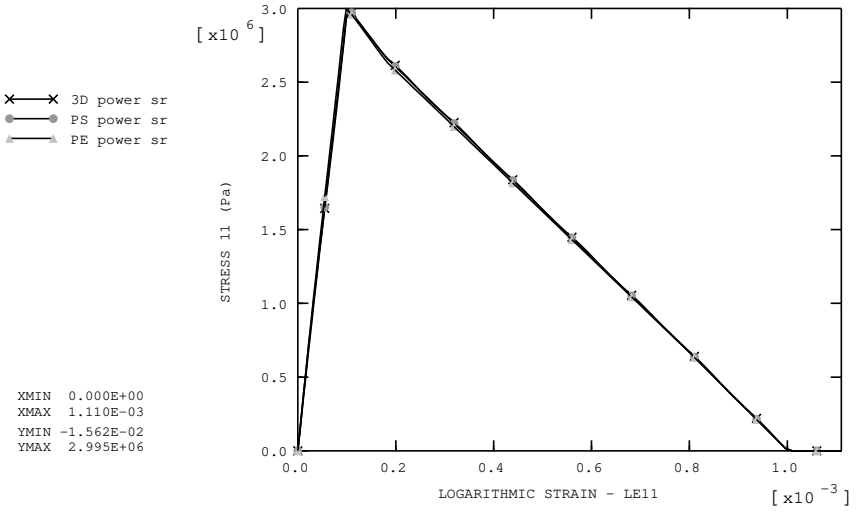


Figure 3: Horizontal stress-strain; power law shear retention.

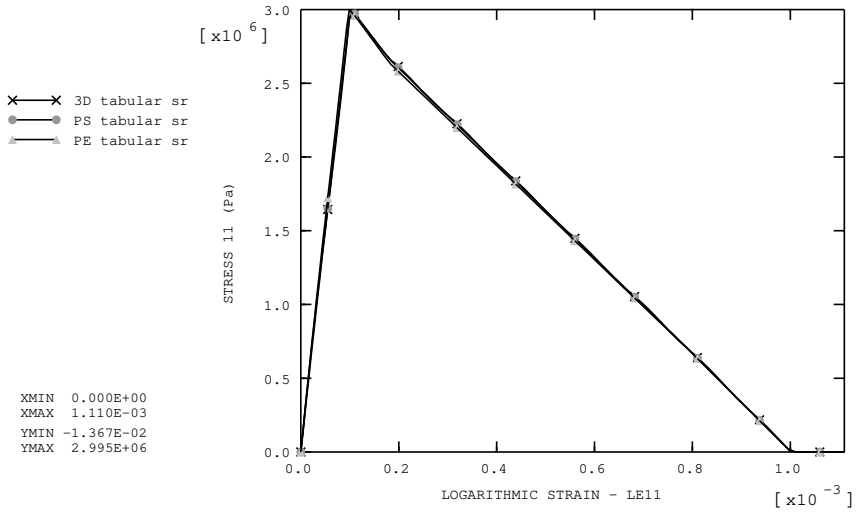


Figure 4: Horizontal stress-strain; tabular shear retention.

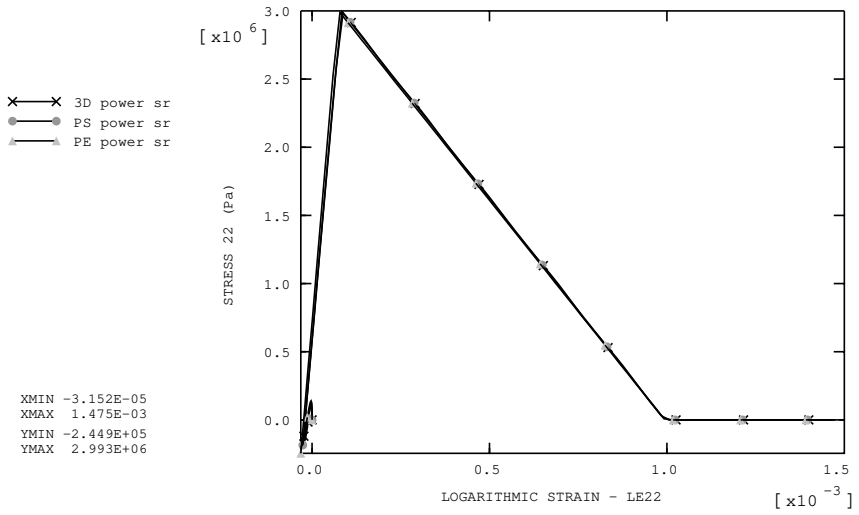


Figure 5: Vertical stress-strain; power law shear retention.

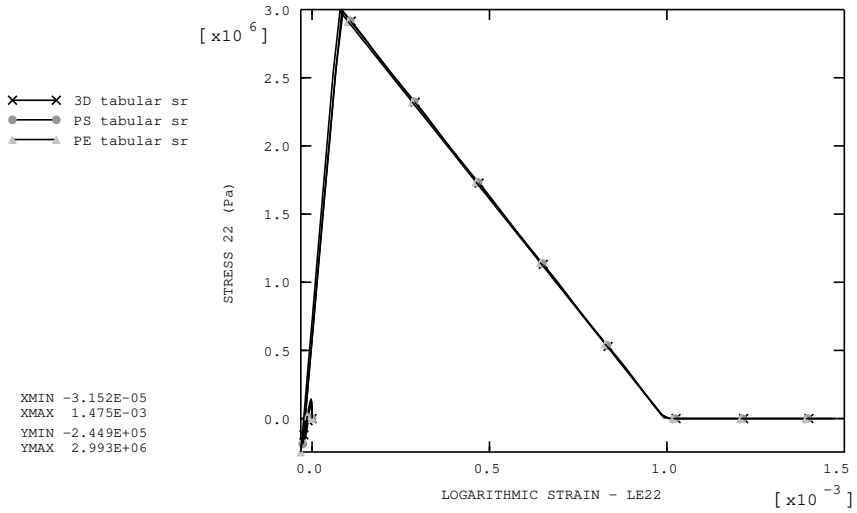


Figure 6: Vertical stress-strain; tabular shear retention.

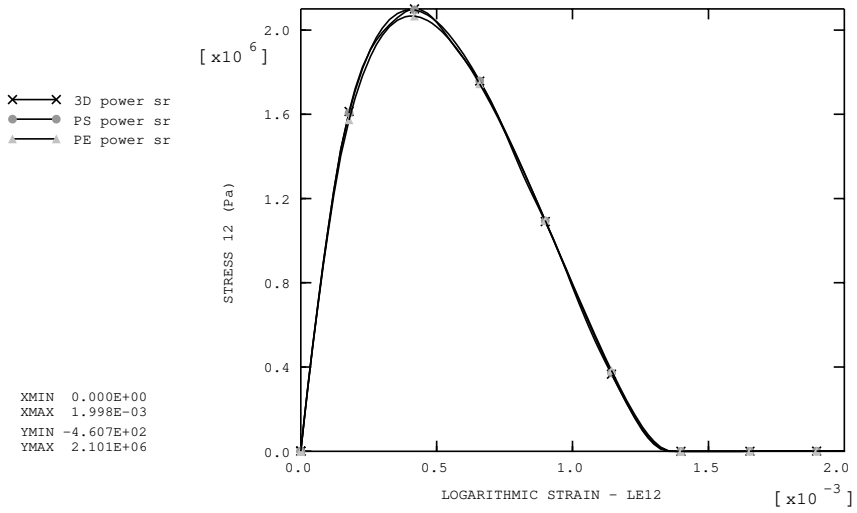


Figure 7: Shear stress-strain; power law shear retention.

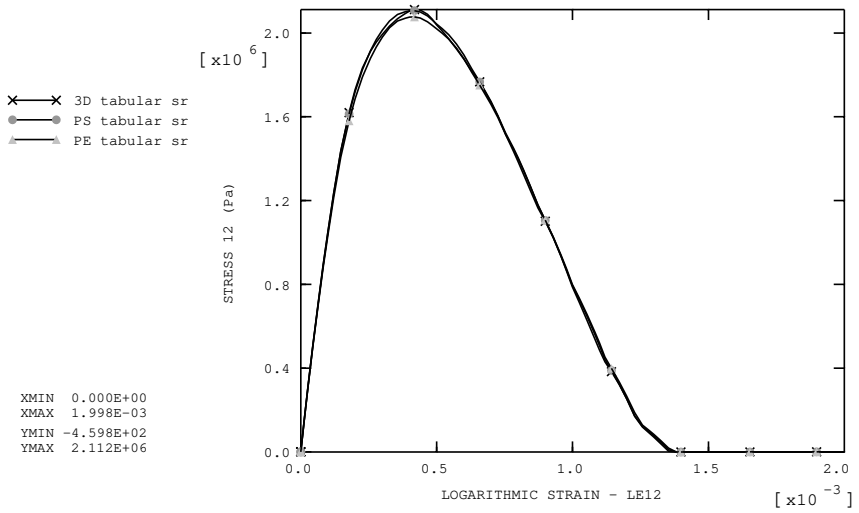


Figure 8: Shear stress-strain; tabular shear retention.

Hydrostatic fluid

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Features tested

This section provides basic verification tests for the fluid behavior associated with fluid elements that are generated in Abaqus/Standard when the fluid cavity capability is used.

Elements tested

F2D2 F3D4

Problem description

Material:

Incompressible fluid

Reference density, ρ_R = 10.0

Reference temperature, θ_0 = 100.

Thermal expansion:

Expansion Coeff.	Temperature
1.E–5	0
4.E–5	300.

Initial temperature, θ_I = 200.

Nearly incompressible fluid

Reference density, ρ_R = 10.0

Bulk compressibility:

Bulk modulus	Temperature
4.E+5	0.
1.E+5	300.

Reference temperature, θ_0 = 100.

Thermal expansion:

Expansion Coeff.	Temperature
1.E–5	0.

Expansion Coeff.	Temperature
4.E–5	300.

Initial temperature, θ_I = 200.

Pneumatic fluid

Ambient pressure, p_A = 14.7

Absolute zero temperature, θ^Z = –460.

Molecular weight, MW = 448.98

Universal gas constant, $\tilde{R}$ = 0.

(The units are not important.)

Results and discussion

The tests in this section are set up as cases of homogeneous deformation of a “block” of hydrostatic fluid. The fluid pressure, temperature, and cavity volume at the cavity reference node are the results of interest. The pressure reported for the pneumatic fluid is the gauge pressure, not the absolute pressure.

The following five steps are executed:

1. Load
2. Increase fluid temperature
3. Add prescribed amount of fluid
4. Remove prescribed amount of fluid
5. Decrease fluid temperature

Input files

mfhhit1ahc.inp	Three-dimensional cavity with incompressible fluid.
mfhhnt1ahc.inp	Three-dimensional cavity with nearly incompressible fluid.
mfhpno1ahc.inp	Three-dimensional cavity with pneumatic fluid.

Composite, mass proportional, and rotary inertia proportional damping in Abaqus/Standard

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Composite damping

Elements tested

B31 MASS ROTARYI SPRING2

Problem description

An eigenvalue analysis is performed on the system consisting of spring, mass, and rotary inertia elements. The spring element builds the stiffness for the translational degrees of freedom, while the mass is assigned to all six degrees of freedom (due to the mass and rotary inertia elements). To avoid solver singularities, a B31 element with negligible mass is included in the model. Composite damping values are also specified for both the mass and rotary inertia elements.

Results and discussion

Since the system is very simple, it is easy to check the value of composite damping per mode. This will be the sum of the element masses times their composite damping values projected into the mode and normalized with the generalized mass of this mode. The composite damping for each of the six requested modes will be equal to 0.01 for the values given in the input file for this test.

Input files

mdacmo1yfr.inp	Input file for composite damping, also tests **LOAD CASE*.
mdacmo1yfr_anis_mass.inp	Input file for composite damping with anisotropic mass; in addition, tests **LOAD CASE*.

Mass proportional damping

Elements tested

MASS SPRING1

Problem description

The linear behavior of a simple spring/mass system with mass proportional damping is tested (see system A in *Linear behavior of spring and dashpot elements*). The MASS element ($m = 0.02588$) is attached to a SPRING1 element; therefore, the system is grounded. The value of the mass proportional damping parameter ($\alpha = 4.6367852$) was taken such that the damping in the system ($c = \alpha m$) is the same as in Problem I in *Linear behavior of spring and dashpot elements* when a dashpot element ($c = 0.12$) is used to provide damping.

Reference solution

Force balance on the system yields a second-order linear differential equation for a single degree of freedom damped oscillator whose solution is identical to the one presented in Problem I in *Linear behavior of spring and dashpot elements*.

Results and discussion

The results for the displacements of the mass match those in *Linear behavior of spring and dashpot elements*.

Input files

mdampo1ydy.inp	Input file for mass proportional damping.
mdampo1ydy_anis_mass.inp	Input file for mass proportional damping with anisotropic mass.

Rotary inertia proportional damping

Elements tested

MASS R2D2 ROTARYI SPRING2

Problem description

The behavior of a simple spring/rigid body system with rotary inertia proportional damping is tested. A rigid body (one R2D2 element), with rotary inertia at its reference node and rotary inertia proportional damping, is allowed only rotation about the z-axis. The rotation of the rigid element is constrained by the two springs acting normal to it. In the first step the rigid body is rotated by 10° in a static procedure, thus developing forces in the springs. In the next dynamic step the above single degree of freedom system is allowed to oscillate freely. An additional perturbation step is included to test the load case definition.

Reference solution

Moment balance on the system yields a second-order linear differential equation for a single degree of freedom damped oscillator. The analytical exponentially decaying sinusoidal solution is obtained for the rotation of the rigid body.

Results and discussion

The results for the rotation of the rigid body match the analytical solution.

Input files

rotary_inertia_damping.inp	Input file for rotary inertia proportional damping, perturbation step with **LOAD CASE*.

Material damping

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

CPS4R CPE4R C3D8R CAX4R B21 B22 B31 B32 M3D4R S4R SAX1

Features tested

Stiffness proportional material damping and band-limited damping.

Problem description

This example problem is used to verify stiffness proportional material damping. A one-dimensional wave is propagated through a single row of elements and allowed to attenuate over time. Both continuum and structural elements are used. The C3D8R element model is shown in *Figure 1*. The row of elements is restrained on one side in the *y*-direction for the two-dimensional element models and restrained in the *y*- and *z*-directions for the three-dimensional element models. All the models are free at both ends in the *x*-direction. For the structural elements the loading is in-plane and all the rotational degrees of freedom are fixed. The damping will cause the amplitude and the frequency of the initial pulse to decrease until the internal energy of the system becomes zero and the bar has a constant longitudinal velocity.

Linear elastic, equation of state, and hyperelastic materials are tested. The elastic material has Young's modulus of 4.4122×10^8 N/m^2 (6.4×10^4 lb/in^2), Poisson's ratio of 0.33, and density of 1.069×10^{10} kg/m^3 (1.0×10^3 lb sec^2 in^{-4}). The behavior of the equation of state material is equivalent to that of the linear elastic material. The elastic shear modulus is 1.6589×10^8 N/m^2 (2.4060×10^4 lb/in^2). For the tabular equation of state material model, the functions are defined as $f_1(\varepsilon_{vol}) = -K\varepsilon_{vol}$ and $f_2(\varepsilon_{vol}) = 0.0$, where $K = 4.3261 \times 10^8$ N/m^2 (6.2745×10^4 lb/in^2). The linear $U_s - U_p$ type of equation of state material has $c_0 = 2.0120 \times 10^{-1}$ m/sec (7.9212 in/sec). The hyperelastic material is a Mooney-Rivlin material, with the constants (for the polynomial strain energy function) $C_{10} =$ 551.6 kPa (80.0 lb/in^2), $C_{01} =$ 137.9 kPa (20 lb/in^2), and $D_1 = 4.5322 \times 10^{-3}$ kPa^{-1} (0.03125 psi^{-1}). Its density is 1.069×10^7 kg/m^3 (1.0 lb sec^2 in^{-4}). In both cases the densities have been increased to slow the wave speed down so that the wavelength of the stress pulse is just shorter than the length of the bar.

The stiffness proportional damping coefficient for both materials is 0.01. A variable stiffness proportional damping can also be defined by specifying the damping coefficient as a tabular function of temperature and/or field variables in Abaqus/Explicit. A large damping coefficient is chosen to illustrate clearly the effects of material damping. In general, this material property is meant to model low level damping of the system, in which case the value of the damping coefficient will be much smaller. In all cases the linear and quadratic bulk viscosities are set equal to zero. This isolates the effects of the stiffness proportional damping.

The models are also used to test band-limited damping in both Abaqus/Standard and Abaqus/Explicit. In all cases, the stiffness proportional damping is replaced with band-limited damping. The damping parameters are chosen as the damping ratio $\eta_d =$ =0.102525, the high-frequency cutoff $f_h = 10$ (Hz), and the low-frequency cutoff $f_l = 4$ (Hz).

Results and discussion

The time history of the energies for the C3D8R element model is shown in *Figure 2*. The value of ALLVD represents the amount of energy lost due to damping. When the stress pulse is between the ends of the bar, the kinetic and strain energies are equal. When a stress wave hits a free surface, the wave is reflected and its sign is reversed. Therefore, when the first half of the wave has hit the free end, the wave that it reflects exactly cancels the tail end of the original wave. At this point all the strain energy in the system has been converted to kinetic energy. Once the wave completely reflects off the end, half of the kinetic energy is transferred back to strain energy. As expected, the wave amplitude decreases. All other element types tested produce similar results.

This problem tests stiffness proportional material damping and band-limited damping for all the available material models, but it does not provide independent verification.

Input files

Abaqus/Standard input files

damp3d_band_std.inp	Three-dimensional solid elements, elastic material definition, band-limited damping, implicit dynamic analysis.
damppe_band_std.inp	Plane strain elements, elastic material definition, band-limited damping, implicit dynamic analysis.
dampps_band_std.inp	Plane stress elements, elastic material definition, band-limited damping, implicit dynamic analysis.
dampax_band_std.inp	Axisymmetric elements, elastic material definition, band-limited damping, implicit dynamic analysis.
dampshell_band_std.inp	Shell elements, elastic material definition, band-limited damping, implicit dynamic analysis.
dampmembrane_band_std.inp	Membrane elements, elastic material definition, band-limited damping, implicit dynamic analysis.
dampbeam2d_band_std.inp	Two-dimensional beam elements, elastic material definition, band-limited damping, implicit dynamic analysis.
dampbeam3d_band_std.inp	Three-dimensional beam elements, elastic material definition, band-limited damping, implicit dynamic analysis.
damptruss2d_band_std.inp	Two-dimensional truss elements, elastic material definition, band-limited damping, implicit dynamic analysis.
damptruss3d_band_std.inp	Three-dimensional truss elements, elastic material definition, band-limited damping, implicit dynamic analysis.
damp3dhyper_band_std.inp	Three-dimensional solid elements, hyperelastic material definition, band-limited damping, implicit dynamic analysis.
damppehyper_band_std.inp	Plane strain elements, hyperelastic material definition, band-limited damping, implicit dynamic analysis.
damppshyper_band_std.inp	Plane stress elements, hyperelastic material definition, band-limited damping, implicit dynamic analysis.
dampaxhyper_band_std.inp	Axisymmetric elements, hyperelastic material definition, band-limited damping, implicit dynamic analysis.
dampshellhyper_band_std.inp	Shell elements, hyperelastic material definition, band-limited damping, implicit dynamic analysis.
dampmembranehyper_band_std.inp	Membrane elements, hyperelastic material definition, band-limited damping, implicit dynamic analysis.

Abaqus/Explicit input files

damp3d.inp	Three-dimensional solid elements, elastic material definition.
damppe.inp	Plane strain elements, elastic material definition.
dampps.inp	Plane stress elements, elastic material definition.
dampax.inp	Axisymmetric elements, elastic material definition.
dampshell.inp	Shell elements, elastic material definition.
dampmembrane.inp	Membrane elements, elastic material definition.
dampbeam2d.inp	Two-dimensional beam elements, elastic material definition.
dampbeam3d.inp	Three-dimensional beam elements, elastic material definition.
damptruss2d.inp	Two-dimensional truss elements, elastic material definition.
damptruss3d.inp	Three-dimensional truss elements, elastic material definition.
damp3deostab.inp	Three-dimensional solid elements, tabulated equation of state material definition.
damp3deosusup.inp	Three-dimensional solid elements, linear $U_s - U_p$ equation of state material definition.
damp3dhyper.inp	Three-dimensional solid elements, hyperelastic material definition.
damppehyper.inp	Plane strain elements, hyperelastic material definition.
dampppshyper.inp	Plane stress elements, hyperelastic material definition.
dampaxhyper.inp	Axisymmetric elements, hyperelastic material definition.
dampshellhyper.inp	Shell elements, hyperelastic material definition.
dampmembranehyper.inp	Membrane elements, hyperelastic material definition.
damp3d_band.inp	Three-dimensional solid elements, elastic material definition, band-limited damping.
damppe_band.inp	Plane strain elements, elastic material definition, band-limited damping.
dampps_band.inp	Plane stress elements, elastic material definition, band-limited damping.
dampax_band.inp	Axisymmetric elements, elastic material definition, band-limited damping.
dampshell_band.inp	Shell elements, elastic material definition, band-limited damping.
dampmembrane_band.inp	Membrane elements, elastic material definition, band-limited damping.
dampbeam2d_band.inp	Two-dimensional beam elements, elastic material definition, band-limited damping.
dampbeam3d_band.inp	Three-dimensional beam elements, elastic material definition, band-limited damping.
damptruss2d_band.inp	Two-dimensional truss elements, elastic material definition, band-limited damping.
damptruss3d_band.inp	Three-dimensional truss elements, elastic material definition, band-limited damping.
damp3deostab_band.inp	Three-dimensional solid elements, tabulated equation of state material definition, band-limited damping.
damp3deosusup_band.inp	Three-dimensional solid elements, linear $U_s - U_p$ equation of state material definition, band-limited damping.
damp3dhyper_band.inp	Three-dimensional solid elements, hyperelastic material definition, band-limited damping.

damppehyper_band.inp	Plane strain elements, hyperelastic material definition, band-limited damping.
damppshyper_band.inp	Plane stress elements, hyperelastic material definition, band-limited damping.
dampaxhyper_band.inp	Axisymmetric elements, hyperelastic material definition, band-limited damping.
dampshellhyper_band.inp	Shell elements, hyperelastic material definition, band-limited damping.
dampmembranehyper_band.inp	Membrane elements, hyperelastic material definition, band-limited damping.

Figures



Figure 1: C3D8R element model.

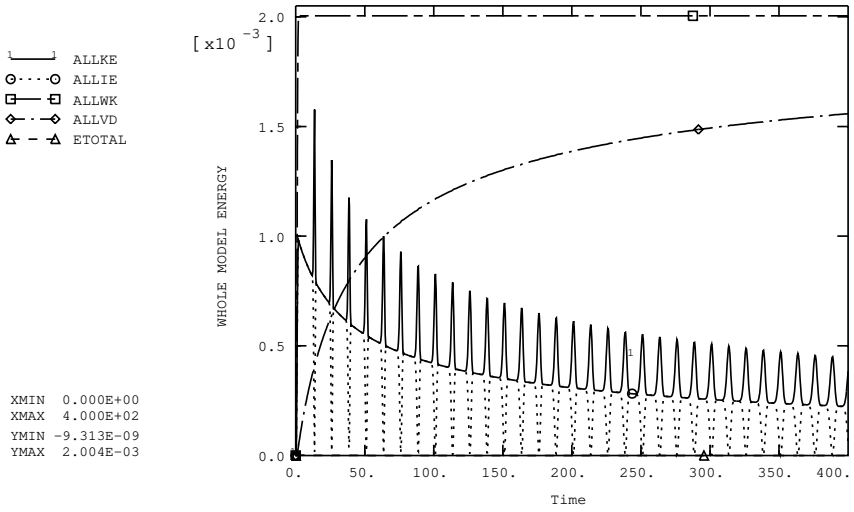


Figure 2: Energy balance as a function of time for three-dimensional continuum elements (C3D8R).

Mass proportional damping

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

T3D2

Features tested

Mass proportional damping.

Problem description

This example is intended to verify mass proportional damping by comparing the Abaqus results with an exact solution for a simple problem.

Mass proportional damping is defined by including damping in the material definition for those elements in which mass proportional damping is required.

The example is the simplest dynamic system: a massless truss connecting a point mass to ground. The mass is obtained by giving the material in the truss a density so that the lumped mass of the truss gives the correct point mass at the free end of the truss. The truss is stretched initially and then relaxed so that it undergoes vibrations of small amplitude. The solution is compared with the exact solution obtained by solving the equation of motion analytically.

Figure 1 shows the geometry. The model consists of a single truss element, type T3D2, constrained at one node and free to move only in the x-direction at its other node. The truss's mass matrix is lumped; therefore, the system is equivalent to a spring and a lumped mass. The cross-sectional area of the truss is 645 mm^2 (1 in^2), and its length is 254 mm (10 in). It is made of linear elastic material, with a Young's modulus of 69 GPa (10^7 lb/in^2). The density of the truss provides a lumped mass at the unrestrained end of 2.777×10^5 kg (1585 $lb\text{-}s^2/in$).

The free end is stretched to displace the mass by 25.4 mm (1 in) in the first step and then released in the second step. The time histories are plotted and compared with the theoretical value.

Results and discussion

The equation of motion for the system is

$$m\ddot{u} + c\dot{u} + ku = 0,$$

where m is the mass, $c\ (= \alpha m)$ is the damping, α is the mass damping factor, k is the stiffness, and u is the displacement.

Assuming a solution of the form $u(t) = u_0 \exp \lambda t$, we have

$$\lambda = \frac{-c}{2m} \pm \sqrt{\left(\frac{c}{2m}\right)^2 - \omega_u^2},$$

where $\omega_u = \sqrt{k/m}$ is the undamped frequency of vibration. Critical damping occurs when the value of c causes the discriminant of this equation to be zero so that

$$c_c = 2m\omega_u = 2\sqrt{km}.$$

We define the damping ratio, ξ, as the ratio of damping to critical damping:

$$\xi = \frac{c}{c_c} = \frac{\alpha}{2\omega_u}.$$

The relationships in this equation are often used as a basis for choosing α and β.

The equation defining λ can be rewritten

$$\lambda = \omega_u \left(-\xi \pm \sqrt{\xi^2 - 1}\right).$$

We choose the damping in this case to be less than critical, $\xi = 0.02 < 1$, and the system can vibrate. The initial conditions are $u(0) = 1$ and $\dot{u}(0) = 0$; therefore, the dynamic part of the motion is

$$u(t) = \exp(-\xi\omega_u t)\left(\frac{\xi}{\sqrt{1-\xi^2}}\sin\omega_d t + \cos\omega_d t\right),$$

where $\omega_d = \omega_u\sqrt{1-\xi^2}$ is the damped frequency of the system.

To generate damping with $\xi = 0.2$, a mass proportional damping factor of $\alpha = 1.00472$ sec^{-1} is used. A variable mass proportional damping can also be defined by specifying the damping coefficient as a tabular function of temperature, field variables, or both in Abaqus. The parameters used in the theoretical results can be calculated as $\omega_u = 25.11802$ rad/sec, $\omega_d = 25.11300$ rad/sec, and

$$u(t) = \exp(-0.5023603t)\,(0.020004001\ \sin 25.113t + \cos 25.113t).$$

The displacement value at the end of Step 2 (t=2.5 sec) is 0.2841910 inches; Abaqus/Explicit gives 0.2717 inches with a 4% relative error, and Abaqus/Standard gives 0.2825 inches with a 0.6% relative error. In the Abaqus/Explicit model, direct user control of the incrementation through the dynamic step is used to achieve smooth and accurate results. The displacement histories predicted by Abaqus/Standard and Abaqus/Explicit are compared with the analytical result in *Figure 2*.

Input files

massdamping_std.inp	Abaqus/Standard input data used in this analysis.
massdamping.inp	Abaqus/Explicit input data used in this analysis.

Figures

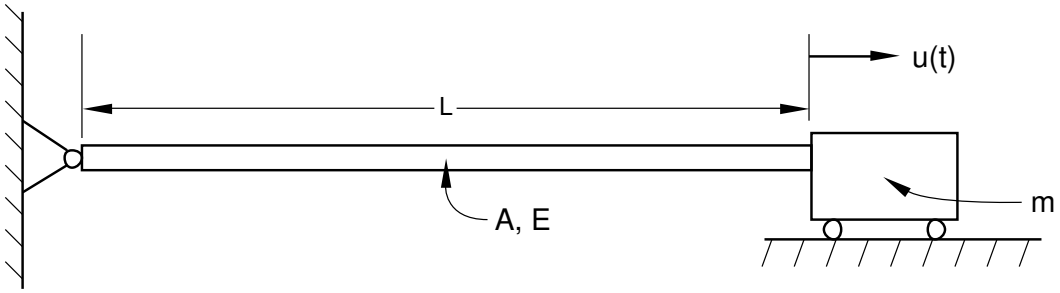


Figure 1: Truss-mass vibration system.

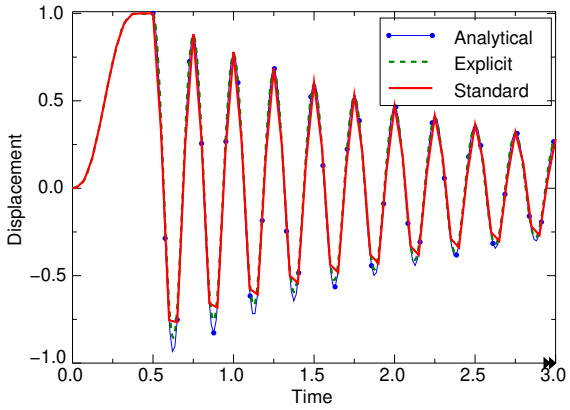


Figure 2: Sample time history.

Structural damping

Products: Abaqus/Standard

References:

- *Structural Damping*

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

B21 B31 PIPE21 PIPE31 T2D2 T3D2
CPE4R CPS4R M3D4R S4R
CAX4R SAX1
C3D8R

Features tested

Structural damping.

Problem description

This example verifies structural damping. A steady-state dynamic analysis is conducted to determine the steady-state response of 1-element structures subjected to uniaxial harmonic excitation. To test constant and frequency-dependent structural damping, results for a linear elastic material A1 with structural damping are compared to results for a linear viscoelastic material A2 without structural damping. The viscoelastic properties of material A2 are chosen so that the structural response of each element made of material A2 is identical to that of the corresponding element made of material A1 with structural damping.

Material: The material parameters used for material A1 are given below.

Elasticity

Table 1: Linear elasticity.

E	ν
64000	0.3

Structural damping

Table 2: Constant structural damping coefficients.

s
0.1

Table 3: Frequency-dependent structural damping coefficients.

s	f
0.05	0.5
0.5	4.0

The material parameters used for material A2 are given below.

Viscoelasticity

Table 4: Linear elasticity.

E	ν
64000	0.3

Table 5: Constant Prony series coefficients.

g_1	k_1	f
0.1	0.1	0.5
0.1	0.1	4.0

Table 6: Frequency-dependent Prony series coefficients.

g_1	k_1	f
0.05	0.05	0.5
0.5	0.5	4.0

Results and discussion

The structural response of the elements made of material A1 with structural damping is identical to that of the corresponding elements made of material A2 with the structurally equivalent viscoelasticity. As a representative example, the displacement amplitude for T3D2, S4R, and C3D8R elements is plotted as a function of the input frequency in *Figure 1* for the case of constant structural damping. Analogous results for the case of frequency-dependent structural damping are plotted in *Figure 2*.

Input files

structural_damping.inp	Elasticity with constant structural damping and structurally equivalent viscoelasticity.
freq_dep_structural_damping.inp	Elasticity with frequency-dependent structural damping and structurally equivalent frequency-dependent viscoelasticity.

Figures

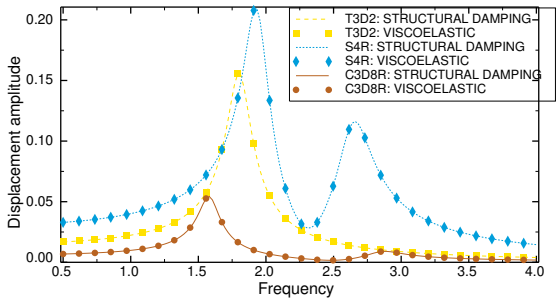


Figure 1: Structural response with constant structural damping and structurally equivalent viscoelasticity.

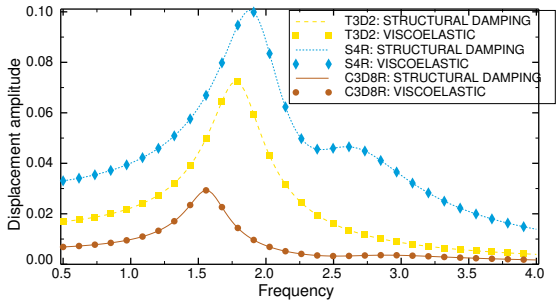


Figure 2: Structural response with frequency-dependent structural damping and structurally equivalent frequency-dependent viscoelasticity.

Thermal expansion test in Abaqus/Standard

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

B31 C3D8R CPE4R CPS4R M3D4R S4R SAX1 T3D2

Features tested

Tangent and secant thermal expansion in Abaqus/Standard.

Problem description

These tests verify the correctness of the thermal strains computed by Abaqus/Standard. Each test consists of a set of elements to which the same elastic-plastic material is assigned. The properties of the material are listed in *Table 1*, *Table 2*, *Table 3*, *Table 4*, *Table 5*, and *Table 6*. The initial temperature in the material is 0°, and it is linearly increased during the analysis to the value of 100°, which generates thermal strains in the models.

Results and discussion

The thermal strain results agree very well with the exact analytical solutions.

Input files

thermal_expansion_elapla_iso_secant.inp	Mises plasticity, isotropic expansion, secant thermal expansion coefficients.
thermal_expansion_elapla_traiso_secant.inp	Mises plasticity, transversely isotropic expansion, secant thermal expansion coefficients.
thermal_expansion_elapla_ortho_secant.inp	Mises plasticity, orthotropic expansion, secant thermal expansion coefficients.
thermal_expansion_elapla_aniso_secant.inp	Mises plasticity, anisotropic expansion, secant thermal expansion coefficients.
thermal_expansion_elapla_iso_tangent.inp	Mises plasticity, isotropic expansion, tangent thermal expansion coefficients.
thermal_expansion_elapla_traiso_tangent.inp	Mises plasticity, transversely isotropic expansion, tangent thermal expansion coefficients.
thermal_expansion_elapla_ortho_tangent.inp	Mises plasticity, orthotropic expansion, tangent thermal expansion coefficients.
thermal_expansion_elapla_aniso_tangent.inp	Mises plasticity, anisotropic expansion, tangent thermal expansion coefficients.

Tables

Table 1: Elastic properties.

E	ν
193.1×10^9	0.3

Table 2: Isotropic plastic hardening.

Yield stress	Plastic strain
206893.0	0.0
413786.0	1.0

Table 3: Isotropic thermal expansion coefficient.

α
10^{-5}

Table 4: Transversely isotropic thermal expansion coefficients.

α_{11}	α_{22}
10^{-5}	5×10^{-6}

Table 5: Orthotropic thermal expansion coefficients.

α_{11}	α_{22}	α_{33}
10^{-5}	5×10^{-6}	2.5×10^{-6}

Table 6: Anisotropic thermal expansion coefficients.

α_{11}	α_{22}	α_{33}	α_{12}
10^{-5}	5×10^{-6}	2.5×10^{-6}	1.2×10^{-6}

Thermal expansion test in Abaqus/Explicit

Products: Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

B21 B22 B31 B32 PIPE21 PIPE31 C3D8R C3D10M CPE4R CPE6M CPS4R CPS6M CAX4R CAX6M M3D4R S4R S4RS S4RSW SAX1 T2D2 T3D2

Features tested

Thermal expansion defined by a predefined temperature field is tested for the following material models: isotropic elasticity, orthotropic elasticity, anisotropic elasticity, lamina, hyperelasticity with polynomial and Ogden forms, hyperelasticity with Arruda-Boyce and Van der Waals forms, hyperfoam, Mises plasticity, Drucker-Prager plasticity, Hill's potential plasticity, crushable foam plasticity with volumetric hardening, crushable foam plasticity with isotropic hardening, ductile failure plasticity, rate-dependent Hill's potential plasticity, rate-dependent Mises plasticity, Drucker-Prager/Cap plasticity, porous metal plasticity, visco-hyperelasticity with polynomial and Ogden forms, visco-hyperelasticity with Arruda-Boyce and Van der Waals forms, and visco-hyperfoam.

Problem description

The verification tests consist of a set of single-element tests that include a combination of all the available elements with all the available materials. All elements are loaded by ramping up the temperature from an initial value of 0° to a final value of 100°. The undeformed meshes are shown in *Figure 1* for the elasticity models, *Figure 2* for the inelasticity models, and *Figure 3* for the viscoelasticity models. Material properties are listed in *Table 1* for the elastic materials and in *Table 2* for the inelastic materials. The thermal expansion coefficient for all materials is 0.00005.

The degrees of freedom in the vertical direction are constrained for all the nodes, and deformation is allowed only in the horizontal direction. Nodes associated with elements C3D8R and C3D10M are constrained in the out-of-plane direction, which causes a plane strain condition to apply for these elements.

Results and discussion

The time history plots for isotropic elasticity, Mises plasticity, and viscoelasticity for all of the elements are shown in *Figure 4*, *Figure 5*, and *Figure 6*, respectively, except for pipe elements, whose results are consistent with beam elements.

Input files

simple_expansion_one.inp	Mises plasticity test.
simple_expansion.inp	Other material models and elements.

Tables

Table 1: Material properties for elastic materials.

Material	Properties	Value
Isotropic elasticity (density=8032)	E	193.1×10^9
	ν	0.3
Orthotropic elasticity (density=7850) (ENGINEERING CONSTANTS)	E_1	2.0×10^{11}
	E_2	1.0×10^{11}
	E_3	1.0×10^{11}
	ν_{12}	0.3
	ν_{13}	0.23
	ν_{23}	0.34
	G_{12}	7.69×10^{10}
	G_{13}	7.69×10^{10}
	G_{23}	9.0×10^9
Orthotropic elasticity (density=7850) (ORTHOTROPIC)	D_{1111}	2.24×10^{11}
	D_{1122}	1.23×10^{11}
	D_{2222}	4.79×10^{11}
	D_{1133}	4.21×10^{10}
	D_{2233}	4.74×10^{10}
	D_{3333}	1.21×10^{11}
	D_{1212}	7.69×10^{10}
	D_{1313}	7.69×10^{10}
	D_{2323}	9.00×10^9
Lamina (density=7800)	E_1	2.0×10^{11}
	E_2	1.5×10^{11}
	ν_{12}	0.35
	G_{12}	2.00×10^{10}
	G_{13}	9.00×10^9
	G_{23}	8.50×10^9

Material	Properties	Value
Foam hyperelasticity (density=0.001)	N	2
	ν	0.01
	uniaxial test	(−0.0217, −0.05)
		... ...
		(−0.02896, −0.80)
	simple shear test	(0.0140, 0.08, 0.0046)
		... ...
		(0.2987, 0.72, 0.1904)
Anisotropic elasticity (density=7850)	D_{1111}	2.24×10^{11}
	D_{1122}	1.23×10^{11}
	D_{2222}	4.79×10^{11}
	D_{1133}	4.21×10^{10}
	D_{2233}	4.74×10^{10}
	D_{3333}	1.21×10^{11}
	D_{1112}	1.00×10^{6}
	D_{2212}	2.00×10^{6}
	D_{3312}	3.00×10^{6}
	D_{1212}	7.69×10^{10}
	D_{1113}	4.00×10^{6}
	D_{2213}	5.00×10^{6}
	D_{3313}	6.00×10^{6}
	D_{1213}	7.00×10^{6}
	D_{1313}	7.69×10^{10}
	D_{1123}	8.00×10^{6}
	D_{2223}	9.00×10^{6}
	D_{3323}	1.00×10^{7}
	D_{1223}	1.10×10^{6}
	D_{1323}	1.20×10^{6}
	D_{2323}	9.00×10^{9}
Polynomial hyperelasticity (density=1000)	N	2
	uniaxial test	(155060, 0.1338)

Material	Properties	Value
		... ...
		$(6.424 \times 10^6, 6.6433)$
	biaxial test	(93840, 0.02)
		... ...
		$(2.465 \times 10^6, 3.45)$
	planar test	(60000, 0.0690)
		... ...
		$(1.82 \times 10^6, 4.0621)$
Ogden hyperelasticity (density=1000)	N	3
	uniaxial test	(155060, 0.1338)
		... ...
		$(6.424 \times 10^6, 6.6433)$
	biaxial test	(93840, 0.02)
		... ...
		$(2.465 \times 10^6, 3.45)$
	planar test	(60000, 0.0690)
		... ...
		$(1.82 \times 10^6, 4.0621)$
Arruda-Boyce hyperelasticity (density=1000)	uniaxial test	(155060, 0.1338)
		... ...
		$(6.424 \times 10^6, 6.6433)$
	biaxial test	(93840, 0.02)
		... ...
		$(2.465 \times 10^6, 3.45)$
	planar test	(60000, 0.0690)
		... ...
		$(1.82 \times 10^6, 4.0621)$
Van der Waals hyperelasticity (density=1000)	uniaxial test	(155060, 0.1338)
		... ...
		$(6.424 \times 10^6, 6.6433)$
	biaxial test	(93840, 0.02)
		... ...
		$(2.465 \times 10^6, 3.45)$
	planar test	(60000, 0.0690)
		... ...
		$(1.82 \times 10^6, 4.0621)$
Viscoelasticity (density=8032)	E	193.1×10^9

Material	Properties	Value
	ν	0.3
	$\overline{g}_i$	0.901001
	$\overline{k}_i$	0.0
	τ_i	0.99
	θ_0	70
	C_1	4.92
	C_2	215
Visco-polynomial hyperelasticity (density=1000)	N	2
	uniaxial test	(155060, 0.1338)
		... ...
		(6.424×10^6, 6.6433)
	biaxial test	(93840, 0.02)
		... ...
		(2.465×10^6, 3.45)
	planar test	(60000, 0.0690)
		... ...
		(1.82×10^6, 4.0621)
	$\overline{g}_i$	0.901001
	$\overline{k}_i$	0.0
	τ_i	0.99
	θ_0	70
	C_1	4.92
	C_2	215
Visco-Ogden hyperelasticity (density=1000)	N	3
	uniaxial test	(155060, 0.1338)
		... ...
		(6.424×10^6, 6.6433)
	biaxial test	(93840, 0.02)
		... ...
		(2.465×10^6, 3.45)
	planar test	(60000, 0.0690)
		... ...
		(1.82×10^6, 4.0621)

Material	Properties	Value
	$\overline{g}_i$	0.901001
	$\overline{k}_i$	0.0
	τ_i	0.99
	θ_0	70
	C_1	4.92
	C_2	215
Visco-foam hyperelasticity (density=0.001)	N	2
	ν	0.0
	uniaxial test	(–0.0217, –0.05)
		... ...
		(–0.02896, –0.80)
	simple shear test	(0.0140, 0.08, 0.0046)
		... ...
		(0.2987, 0.72, 0.1904)
	$\overline{g}_i$	0.901001
	$\overline{k}_i$	0.0
	τ_i	0.99
	θ_0	70
	C_1	4.92
	C_2	215
Visco-Arruda-Boyce hyperelasticity (density=1000)	uniaxial test	(155060, 0.1338)
		... ...
		(6.424×10^6, 6.6433)
	biaxial test	(93840, 0.02)
		... ...
		(2.465×10^6, 3.45)
	planar test	(60000, 0.0690)
		... ...
		(1.82×10^6, 4.0621)
	$\overline{g}_i$	0.901001
	$\overline{k}_i$	0.0
	τ_i	0.99

Material	Properties	Value
	θ_0	70
	C_1	4.92
	C_2	215
Visco-Van der Waals hyperelasticity (density=1000)	uniaxial test	(155060, 0.1338)
		... ...
		(6.424×10^6, 6.6433)
	biaxial test	(93840, 0.02)
		... ...
		(2.465×10^6, 3.45)
	planar test	(60000, 0.0690)
		... ...
		(1.82×10^6, 4.0621)
	$\overline{g}_i$	0.901001
	$\overline{k}_i$	0.0
	τ_i	0.99
	θ_0	70
	C_1	4.92
	C_2	215

Table 2: Material properties for inelastic materials.

Material	Properties	Value
Mises plasticity (density=8032)	E	193.1×10^9
	ν	0.3
	σ_0	206893
	H	206893
Drucker plasticity (density=1000)	E	2.0×10^7
	ν	0.3
	σ_0	40000
	H	40000
	β	40
	K	1.0
	ψ	20.0
Hill's plasticity (density=2500)	E	1.0×10^9

Material	Properties	Value
	ν	0.3
	σ_0	1.0×10^6
	H	4.0×10^5
	R_{11}	1.5
	R_{22}	1.0
	R_{33}	1.0
	R_{12}	1.0
	R_{13}	1.0
	R_{23}	1.0
Crushable foam with volumetric hardening (density=500)	E	3.0×10^6
	ν	0.0
	k	1.1
	k_t	0.1
	hardening	(2.2×10^5, 0.0)
		... ...
		(6.88×10^5, 10.0)
Crushable foam with isotropic hardening (density=500)	E	3.0×10^6
	ν	0.0
	k	1.1
	ν_p	0.2983
	hardening	(2.2×10^5, 0.0)
		... ...
		(6.88×10^5, 10.0)
Ductile failure (density=5800)	E	2.0×10^8
	ν	0.3
	σ_0	2.0×10^5
	H	4.0×10^5
	$\bar{\epsilon}_f^{pl}$	0.5
Mises plasticity (density=8032)(rate dependent)	E	193.1×10^9
	ν	0.3
	σ_0	206893
	H	206893
	D	1000

Material	Properties	Value
	p	2.0
Hill's plasticity (density=2500)(rate dependent)	E	1.0×10^9
	ν	0.3
	σ_0	1.0×10^6
	H	4.0×10^5
	R_{11}	1.5
	R_{22}	1.0
	R_{33}	1.0
	R_{12}	1.0
	R_{13}	1.0
	R_{23}	1.0
	D	4000
	p	6.0
Drucker-Prager/Cap plasticity(density=0.0024)	E	30000
	ν	0.3
	d	100
	β	37.67
	R	0.1
	ϵ_{vol}^{pl}	0.0
	α	0.01
	hardening	(20.96, 0)
		... ...
		(655.6, 0.00249)
Porous metal plasticity(density=7.7×10^7)	E	2.0×10^{11}
	ν	0.33
	σ_0	7.5×10^8
	H	0.0
	q_1	1.0
	q_2	1.25
	q_3	1.0
	ϵ_N	0.1
	s_N	0.06
	f_N	0.04

Material	Properties	Value
	f_F	0.8
	f_c	0.5

Figures

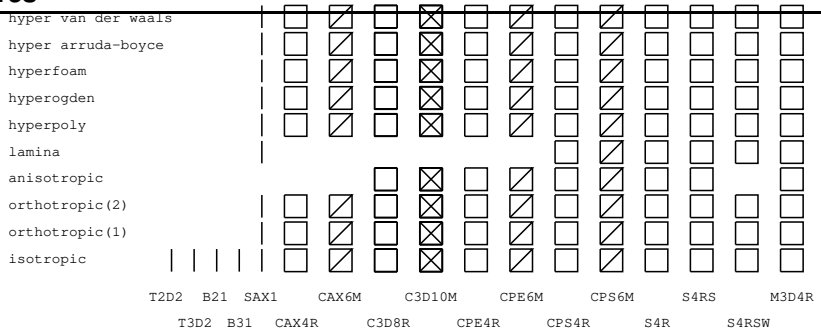


Figure 1: Simple expansion test for elastic materials.

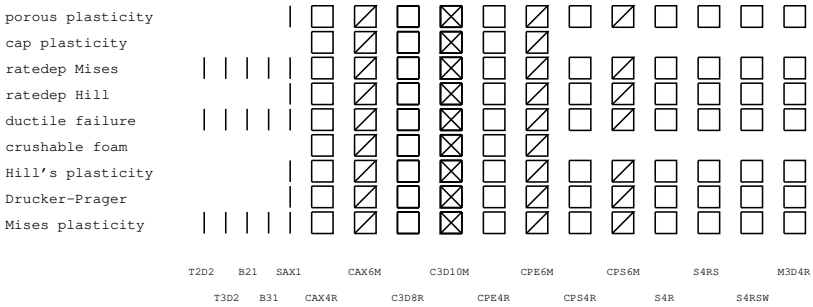


Figure 2: Simple expansion test for inelastic materials.

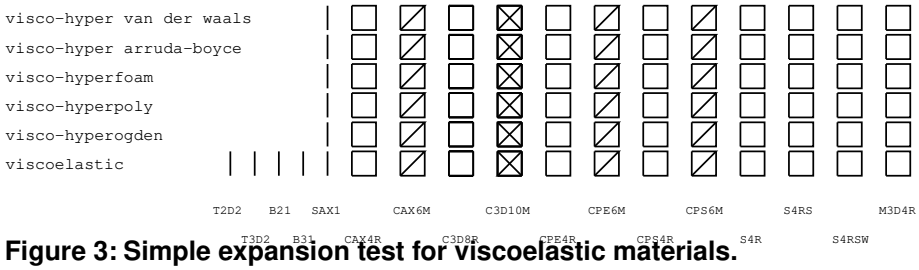


Figure 3: Simple expansion test for viscoelastic materials.

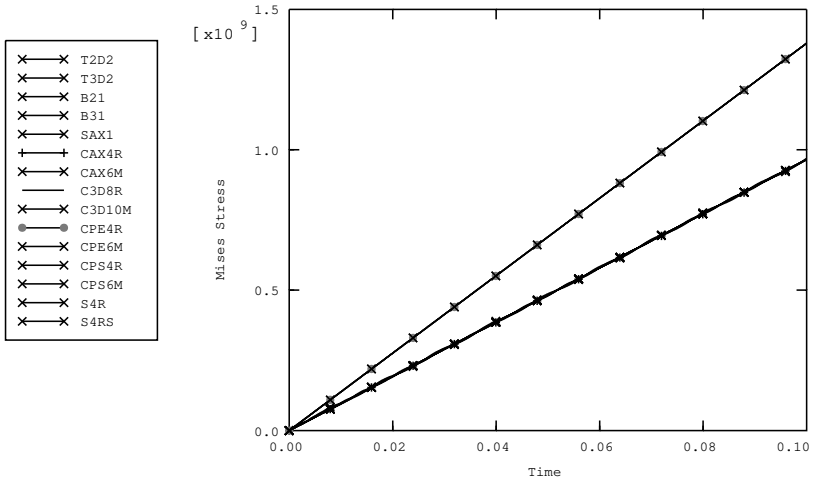


Figure 4: Mises stress versus time for isotropic elasticity.

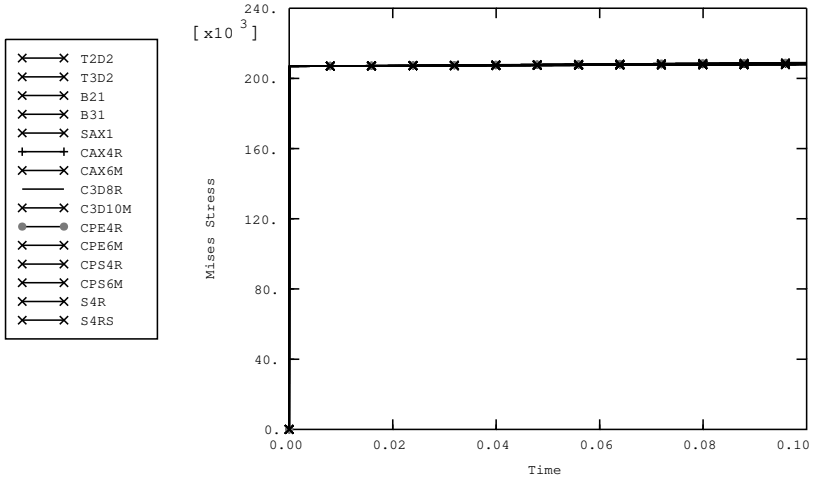


Figure 5: Mises stress versus time for Mises plasticity.

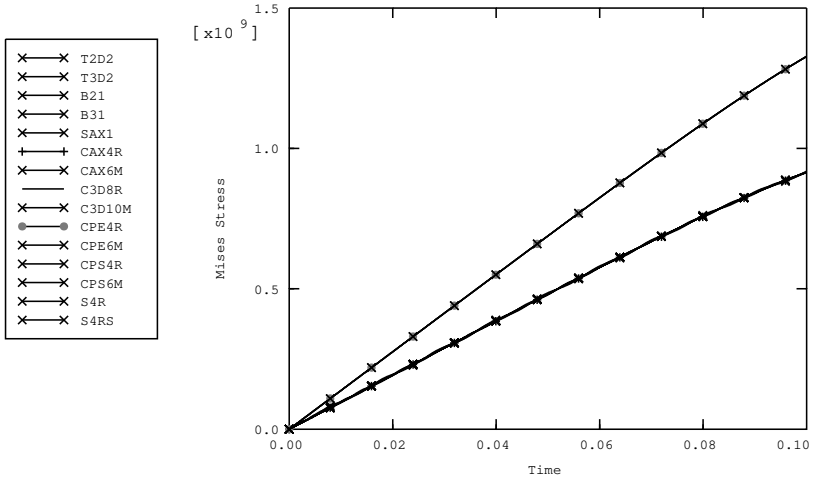


Figure 6: Mises stress versus time for viscoelasticity.

Field expansion test

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D8PT QEC3D8

Problem description

The model consists of a cube of porous material of initial side length L_0 and volume $V_0 = {L_0}^3$, which is subjected to unconstrained thermal and field expansion. The expansion is assumed to be isotropic, and geometric nonlinearity is assumed in all the calculations. Due to stress-free expansion, the overall volume ratio of the porous medium, J, is equal to the volume ratio of the solid grains, J_g, and the porosity, n, stays constant during the expansion. Denoting the logarithmic expansion strains in the solid grains and in the pore fluid as ϵ_s and ϵ_f, respectively, the corresponding stretches are $\lambda_s = \exp(\epsilon_s)$ and $\lambda_s = \exp(\epsilon_f)$, so that the expanded cube side length becomes $L_1 = \lambda_s L_0$, with volume $V_1 = {L_1}^3$. Assuming the porous medium to be fully saturated over the history of the simulation, the initial volume of the pore fluid is $V_0 = {L_0}^3 n$. The expanded fluid volume is $V_{1f} = {L_{1f}}^3 n$, where $L_{1f} = \lambda_f L_0$. The change of the total volume for the pores is $\Delta V_p = (V_1 - V_0)\, n$, while the change of volume of the pore fluid due to fluid expansion is $\Delta V_f = V_{1f} - V_{0f}$. If ϵ_s and ϵ_f are different from each other, the volume of fluid that must enter the model to maintain full saturation is $\Delta V_p - \Delta V_f$. For an 8-node element, the nodal reactive fluid total volume (output variable RVT) is obtained as $(\Delta V_p - \Delta V_f)\,(V_{0f}/V_{1f})/8$, where the fluid volume ratio accounts for expansion of the incoming fluid.

The model has three elements with different material properties. The solid and the pore fluid phases in the elements are subjected to a combination of different sources of expansion (multiple field expansions, as well as thermal expansion), such that the total ϵ_s and ϵ_f are the same in all three elements. *Figure 1* shows the 11-component of the field expansion strain due to field variable number 1.

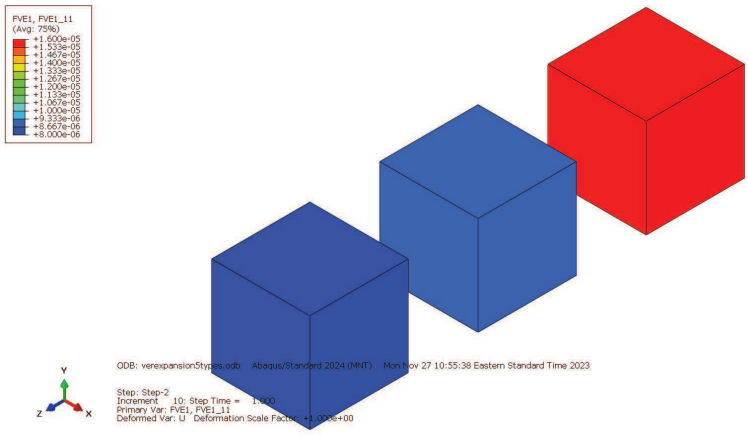


Figure 1: Field expansion strain due to field variable number 1.

Even though the total values of ϵ_s and ϵ_f are the same for all three elements, the sources of expansion are different in each case. For example, only elements 1 and 3 have the field expansion in the pore fluid due to field variable 3, FVEFL3, defined (see *Figure 2*; note that element 2 does not show any contour because it does not contain output variable FVEFL3).

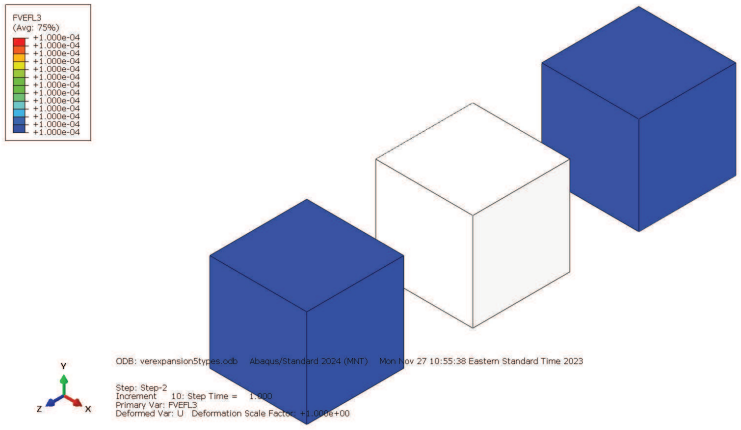


Figure 2: Field expansion strain due to field variable number 3.

Results and discussion

A transient soils consolidation analysis with 10 time increments is carried out. The numerical solutions from Abaqus for the total expansion and RVT agree very well with the analytical solution.

Input files

field_thermal_expansion_porous.inp	Test for C3D8PT elements.
field_thermal_expansion_porous2.inp	Test for QEC3D8 elements.
field_thermal_expansion_porous.f	User subroutine *UEXPAN* for both input files.

Multiscale materials

The macroscale properties of the composites predicted by the mean-field homogenization (MFH) and the deep material network (DMN) methods are compared to the Finite Element-Representative Volume Element (FE-RVE) predictions.

Finite element models of a repeating unit cell use the same constituent properties and microstructure of the composite as the mean-field model. Periodic boundary conditions are applied on the boundary nodes of the unit cell using the following equation:

$$u_i\left(x_j + p_j^{\alpha}\right) = u_i\left(x_j\right) + \langle u_{i,k}\rangle\, p_k^{\alpha},$$

where u_i is the nodal displacement, p_j^{α} is the α^{th} vector defining the periodicity of the unit cell, and $\langle u_{i,k}\rangle$ is the far-field displacement gradient, which is assumed to be equal to the nominal strain given that the far-field rotation is constrained to be zero. The stress response of the FE-RVE is given by

$$\langle \sigma_{ij}\rangle = \frac{1}{V}\int_{\Omega}\sigma_{ij} \qquad\qquad V \simeq \frac{\sum_{k=1,n}\sigma_{ij}^{k}V_k}{\sum_{k=1,n}V_k}.$$

This stress response can then be used to validate the prediction given by the MFH. The FE-RVE can also be used to obtain the stiffness matrix by performing a linear perturbation step with six load cases, with each load case yielding one column of the 6 × 6 matrix. The Coefficient of Thermal Expansions (CTE) can be computed by applying a uniform temperature change across the FE-RVE and measuring the change in the volume-averaged strain.

In this section:

- *Multiscale linear elastic materials*
- *Multiscale linear viscoelastic materials*
- *Multiscale nonlinear materials*
- *Multiscale materials with cure modeling*

Multiscale linear elastic materials

Products: Abaqus/Standard

References:

- *Mean-Field Homogenization*
- *Multiscale Material Modeling*

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D8R

Features tested

Linear elastic multiscale materials modeled with MFH methods and DMN.

Problem description

The verification tests consist of a set of single-element tests that use C3D8R elements under mechanical and thermal loading conditions. All constituents of the composite material are linear elastic. Material properties including Young's modulus, E, and Poisson's ratio, ν, are listed below. Microstructure properties including volume fraction, v_f, and aspect ratio, a_r, are also listed.

Material:

Composite 1

Matrix properties:

E	250 GPa
ν	0.3
v_f	80%

Inclusion one properties:

E	750 GPa
ν	0.3
v_f	10%
a_r	infinite

Inclusion two properties:

E	1000 GPa
ν	0.3

v_f	10%
a_r	infinite

Both inclusions are cylindrical and aligned.

Composite 2

Matrix properties:

E	3.16 GPa
ν	0.35

Inclusion properties:

E	73.1 GPa
ν	0.18
a_r	1.0

The inclusions are spherical with variable volume fractions.

Composite 3

Matrix properties:

E	70 GPa
ν	0.33

Inclusion properties:

E	300 GPa
ν	0.2
v_f	10%
a_r	20.0

The shape of the inclusion is prolate, and the inclusions are aligned.

Composite 4

Matrix properties:

E	4.5 GPa
ν	0.38

Inclusion properties:

E	172 GPa
ν	0.2
a_r	0.04

The inclusion is penny-shaped and randomly oriented in 3D.

Composite 5

The material properties are the same as those for Composite 4, but the inclusion directions are specified with a different second-order orientation tensor,

$$a = \begin{bmatrix} 0.515 & -0.038 & -0.059 \\ -0.038 & 0.375 & -0.063 \\ -0.059 & -0.063 & 0.11 \end{bmatrix}.$$

Composite 6

Matrix properties:

E	3 GPa
ν	0.35
CTE	70e-6 K^{-1}

Inclusion properties:

E	172 GPa
ν	0.2
CTE	1e-6 K^{-1}

The volume fraction and aspect ratio of the inclusion are variable.

Composite 7

Matrix properties:

E	3.45 GPa
ν	0.35

Inclusion properties:

E	73.1GPa
ν	0.22

The inclusion is continuous fiber (cylinder-shaped).

Composite 8

Matrix properties:

E	5.35 GPa
ν	0.354

Inclusion properties (transversely isotropic):

E_1	232 GPa
E_2	15 GPa
ν_{12}	0.279
ν_{23}	0.49
G_{12}	24 GPa

The inclusion is continuous fiber (cylinder-shaped).

Results and discussion

A two-step MFH method is used to obtain the transverse response of Composite 1. Both Mori-Tanaka and the balanced homogenization model are used in Step 1, and Voigt homogenization is used in Step 2. Results are compared with the FE-RVE prediction in *Table 1*. The MFH predictions show good agreement with the FE-RVE results.

The influence of the volume fraction on the normalized macroscale modulus of Composite 2 is plotted in *Figure 2* using various homogenization methods. Two different FE-RVEs are used for comparison: body-centered and square packed. The body-centered FE-RVE is shown in *Figure 1*. The square-packed FE-RVE is also a cubic cell with only a single sphere in the middle of the cell. The Mori-Tanaka prediction matches the body-centered FE-RVE prediction very well, while the balanced prediction matches the square-packed FE-RVE results very well up to a volume fraction of 50%. Interaction between inclusions is stronger in the squared-packed FE-RVE compared to the body-centered FE-RVE, which can be better captured with the balanced method.

Composites with various off-axis angles are tested with Composite 3. The influence of the off-axis angle on the normalized macroscale modulus in the traction direction is plotted in *Figure 3*. Mori-Tanaka and the balanced homogenization methods are used, and the predicted macroscale properties compare well with the FE-RVE results. The body-centered FE-RVE model is used to obtain the stiffness modulus of the composite through a linear perturbation step as described above. The engineering constants predicted by the FE-RVE model are listed in *Table 2*, with the 1-direction being the inclusion direction. The axial macroscale elastic modulus is then computed at all off-axis angles through transformation.

The influence of the volume fraction on the macroscale elastic modulus of Composite 4 in the traction direction is plotted in *Figure 4*. 6 × 6 angle subdivisions and 12 × 12 angle subdivisions are used for the numerical integration over the orientations. Two-step homogenization is used; Mori-Tanaka homogenization is used in Step 1, and Voigt homogenization is used in Step 2. For comparison, the macroscale elastic modulus is also computed directly using the orientation average,

$$C_{ijkl} = C_1 a_{ijkl} + C_2 \left(a_{ij}\delta_{kl} + a_{kl}\delta_{ij}\right) + C_3 \left(a_{ik}\delta_{jl} + a_{il}\delta_{jk} + a_{jl}\delta_{ik} + a_{jk}\delta_{il}\right) + C_4\delta_{ij}\delta_{kl} + C_5(\delta_{ik}\delta_{jl} + \delta_{il}\delta_{jk}),$$

where $C_i's$ are the five constants of a transversely orthotropic material and C^{UD}_{ijkl} are the components of the elastic modulus of the transversely orthotropic material with symmetry around axis-1:

$$C_1 = C^{UD}_{1111} - 2C^{UD}_{1133} + C^{UD}_{3322} - 4C^{UD}_{1313} + 2C^{UD}_{2323}$$
$$C_2 = C^{UD}_{1133} - C^{UD}_{3322}$$
$$C_3 = C^{UD}_{1313} - C^{UD}_{2323}$$
$$C_4 = C^{UD}_{3322}$$
$$C_5 = C^{UD}_{2323}.$$

a_{ij} is the second-order orientation tensor, which is

$$a = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

for 3D randomly distributed orientations. The fourth-order orientation tensor is computed using

$a_{ijkl} = \iint p_i p_j p_k p_l \Psi(\theta, \varphi) \sin\theta d\theta d\varphi$, where $p = [\sin\theta\cos\varphi, \sin\theta\sin\varphi, \cos\theta]$ is the fiber direction and Ψ is the orientation distribution function.

The same studies are made with Composite 5, and the results are plotted in *Figure 5*. The predicted macroscale modulus matches well with the orientation average.

Thermal-elastic composites under thermal loading are also tested with Composite 6. The influence of aspect ratio on the predicted macroscale coefficient of thermal expansion (CTE) is plotted in *Figure 6*.

A comparison of homogenized elastic constants is performed for Composites 7 and 8. Different MFH homogenization methods are used. Both composites are UD composites with continuous fibers. The fiber material in Composite 7 is isotropic glass fiber, and the fiber material in Composite 8 is transversely isotropic carbon fiber. The homogenized elastic properties are transversely isotropic, and the 1-direction is the fiber direction, which is also the 1-direction of the local direction of the fiber. We compared the homogenized elastic constants computed using Chamis, CCA, and Mori-Tanaka formulations with FE-RVE results at various volume fractions.

The fibers in the FE-RVEs are hex packed, as shown in *Figure 7*. The homogenized constants E_1 and ν_{12} are very close between all results. Therefore, only E_2 and G_{12} and G_{23} are shown in the figures. Results for epoxy-glass Composite 7 are shown in *Figure 8*, *Figure 10*, and *Figure 12*. We can see that the homogenized results compare well with FE-RVE results, with Chamis and CCA slightly better than Mori-Tanaka at high volume fractions. Results for epoxy-carbon Composite 8 are shown in *Figure 14*, *Figure 16*, and *Figure 18*. The results based on Chamis formulation show the largest deviation compared to FE-RVE results, while both CCA and Mori-Tanaka compare well with FE-RVE results for all volume fractions.

DMN and Mori-Tanaka results are compared against FE-RVE predictions for Composites 7 and 8. *Figure 9*, *Figure 13*, and *Figure 15* show the comparisons of homogenized elastic constants for Composite 7. *Figure 15*, *Figure 17*, and *Figure 19* show the comparisons of homogenized elastic constants for Composite 8. The DMN predictions compare well with FE-RVE. When the volume fraction is higher than the range where the DMN network is trained ($V_f > 65\%$), the DMN predictions deviate from the corresponding Mori-Tanaka results. FE-RVE results are not available for high volume fractions when the hex-packed pattern reaches its limit.

Input files

multscale-twostep-mt.inp	One-element test using Composite 1 with two-step MFH. Mori-Tanaka homogenization is used in the first step, and Voigt homogenization is used in the second step.
multscale-twostep-balanced.inp	One-element uniaxial tension test using Composite 1. Traction is applied in the transverse direction. Two-step MFH is used. Balanced homogenization is used in the first step, and Voigt homogenization is used in the second step.
multscale-onestep.inp	One-element uniaxial tension test using Composite 1. Traction is applied in the transverse direction. Direct Mori-Tanaka homogenization is used, which is equivalent to using Mori-Tanaka homogenization in both the first and second step.
rve-twocylinder.inp	FE-RVE model of Composite 1 under transverse loading.
multscale-sphere-vf.inp	One-element uniaxial tension tests using Composite 2 with various volume fractions and homogenization methods.
multscale-sphere-vf2.inp	One-element uniaxial tension tests using Composite 2 with various volume fractions and homogenization methods.
rve-sphere-vf10-bc.inp	FE-RVE model of Composite 2 with a volume fraction of 10% under uniaxial tension loading. Inclusions are packed with a body-centered pattern.

rve-sphere-vf10-sq.inp	FE-RVE model of Composite 2 with a volume fraction of 10% under uniaxial tension loading. Inclusions are packed with a square pattern.
multscale-angle-mt.inp	One-element uniaxial tension tests using Composite 3 with various off-axis angles specified with fixed inclusion directions. Mori-Tanaka homogenization is used.
multscale-angle-balanced.inp	One-element uniaxial tension tests using Composite 3 with various off-axis angles specified with fixed inclusion directions. Balanced homogenization is used.
rve-prolate-vf10.inp	FE-RVE model of Composite 3 under uniaxial tension loading. Inclusions are packed with a body-centered pattern and aligned in the direction of the traction.
multscale-penny-random3d-6.inp	One-element uniaxial tension test using Composite 4. Mori-Tanaka homogenization is used; 6 × 6 angle divisions are used for numerical integration.
multscale-penny-random3d-12.inp	One-element uniaxial tension test using Composite 4. Mori-Tanaka homogenization is used; 12 × 12 angle divisions are used for numerical integration.
multscale-penny-oritens-6.inp	One-element uniaxial tension test using Composite 5 with dispersed inclusion orientations specified with a second-order orientation tensor. Mori-Tanaka homogenization is used; 6 × 6 angle divisions are used for numerical integration.
multscale-penny-oritens-12.inp	One-element uniaxial tension test using Composite 5 with dispersed inclusion orientations specified with a second-order orientation tensor. Mori-Tanaka homogenization is used; 12 × 12 angle divisions are used for numerical integration.
multscale-cte-vf2.inp	One-element tests using Composite 6 under thermal loading. Various aspect ratios are used for the inclusions. Mori-Tanaka homogenization is used. The volume fraction of the inclusion is 2%.
multscale-cte-vf5.inp	One-element tests using Composite 6 under thermal loading. Various aspect ratios are used for the inclusions. Mori-Tanaka homogenization is used. The volume fraction of the inclusion is 5%.
multscale-cte-vf10.inp	One-element tests using Composite 6 under thermal loading. Various aspect ratios are used for the inclusions. Mori-Tanaka homogenization is used. The volume fraction of the inclusion is 10%.

References

- Pierard, O., C. Friebel, and I. Doghri, "Mean-Field Homogenization of Multi-Phase Thermo-Elastic Composites: a General Framework and its Validation," *Composites Science and Technology*, vol. 64, pp. 1587–1603, 2004.
- Advani, S., and C. Tucker, "The Use of Tensors to Describe and Predict Fiber Orientation in Short Fiber Composites," *Journal of Rheology*, vol. 31(8), pp. 751–784, 1987.
- Onat, E.T., and F. A. Leckie, "Representation of Mechanical Behavior in the Presence of Changing Internal Structure," *Journal of Applied Mechanics*, vol. 55(1), pp. 1–10, 1988.

Tables

Table 1: Comparison of normalized macroscale transverse elastic modulus with different homogenization methods. Prediction is compared with FE-RVE results.

Method	Mori-Tanaka/Voigt	Balanced/Voigt	Mori-Tanaka/Mori-Tanaka	FE-RVE
$\overline{E_T}/E_0$	1.2340	1.2408	1.2340	1.2412

Table 2: Normalized macroscale properties of a body-centered FE-RVE with prolate inclusions. The elastic moduli are normalized with the Young's modulus of the matrix constituent E_0.

E_1	E_2	E_3	ν_{12}	ν_{13}	ν_{23}	G_{12}	G_{13}	G_{23}
1.28	1.130	1.13	0.316	0.316	0.347	0.421	0.421	0.417

Figures

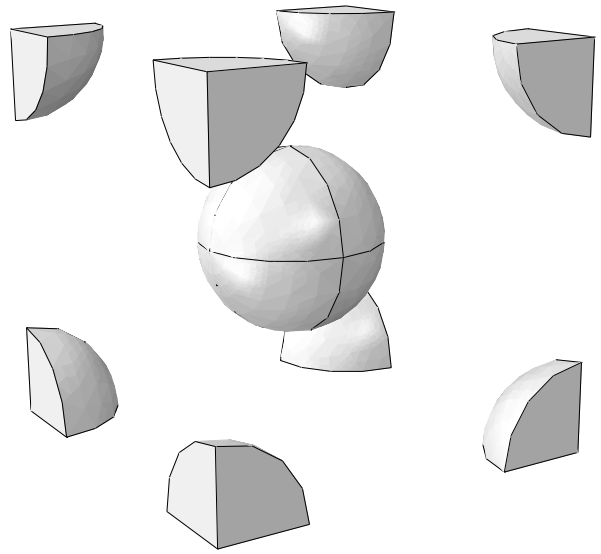


Figure 1: Body-center packed composite with spherical inclusions (matrix is not shown).

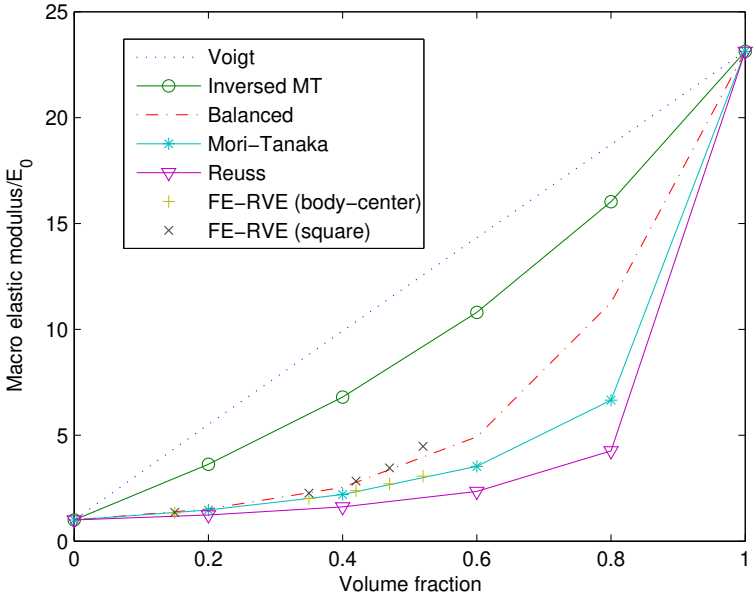


Figure 2: Predicted macroscale elastic modulus with different homogenization methods.

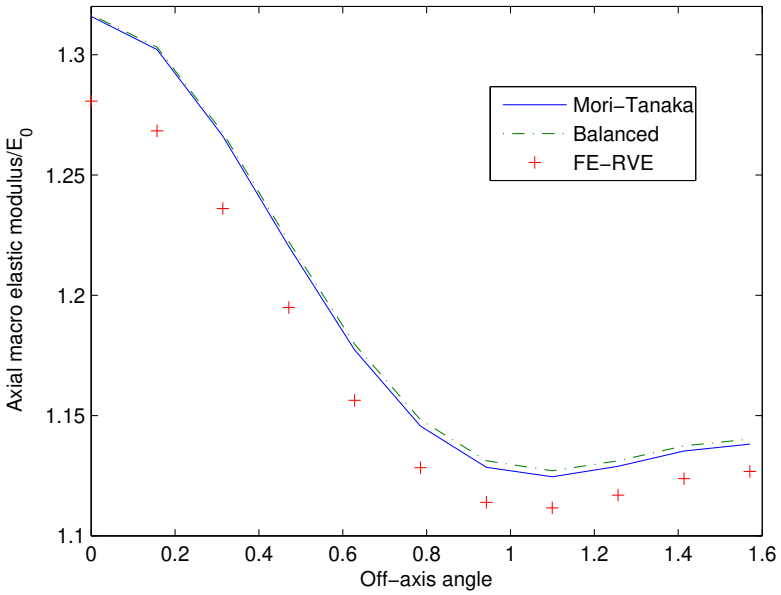


Figure 3: Comparison between predicted axial macroscale elastic modulus at different off-axis angles with MFH methods (Mori-Tanaka and balanced) and FE-RVE results.

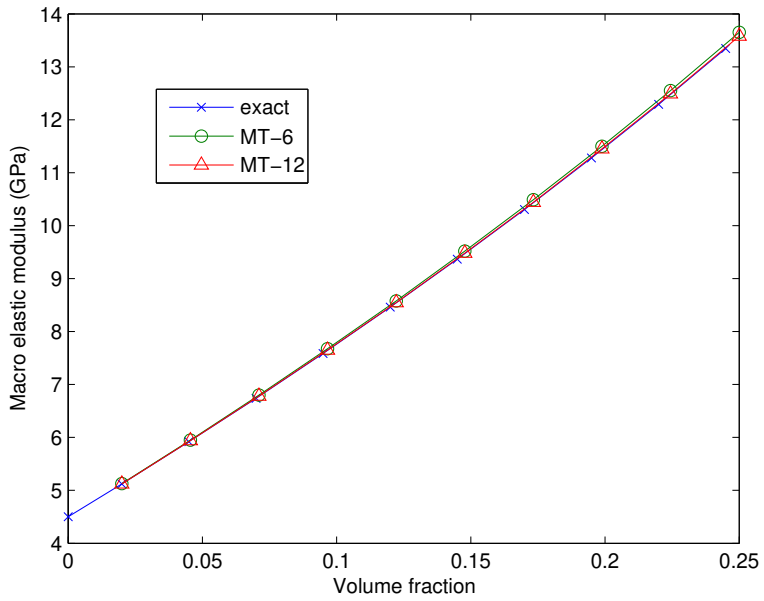


Figure 4: Prediction of the macroscale elastic modulus of a composite with random oriented penny-shaped inclusions at different volume fractions; 6 × 6 grains and 12 × 12 angle divisions are used for the numerical integration using the orientation distribution function, and the results are compared with the orientation average.

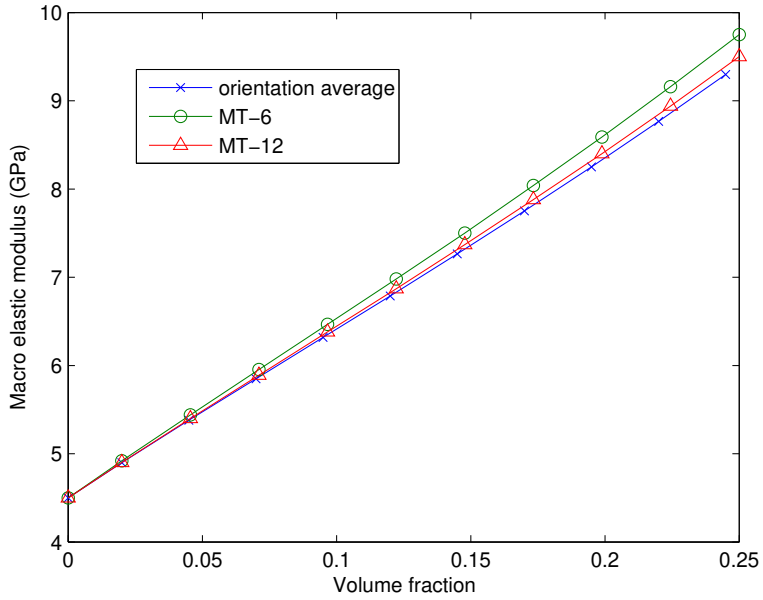


Figure 5: Prediction of the macroscale elastic modulus of a composite with penny-shaped inclusions at different volume fractions. The inclusion directions are specified using a second-order orientation tensor; 6 × 6 grains and 12 × 12 angle divisions are used for the numerical integration using the orientation distribution function, and the results are compared with the orientation average.

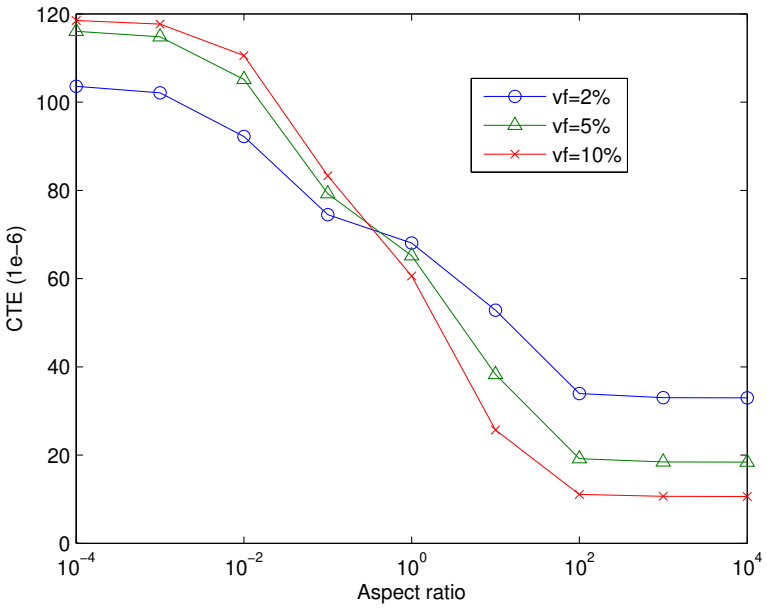
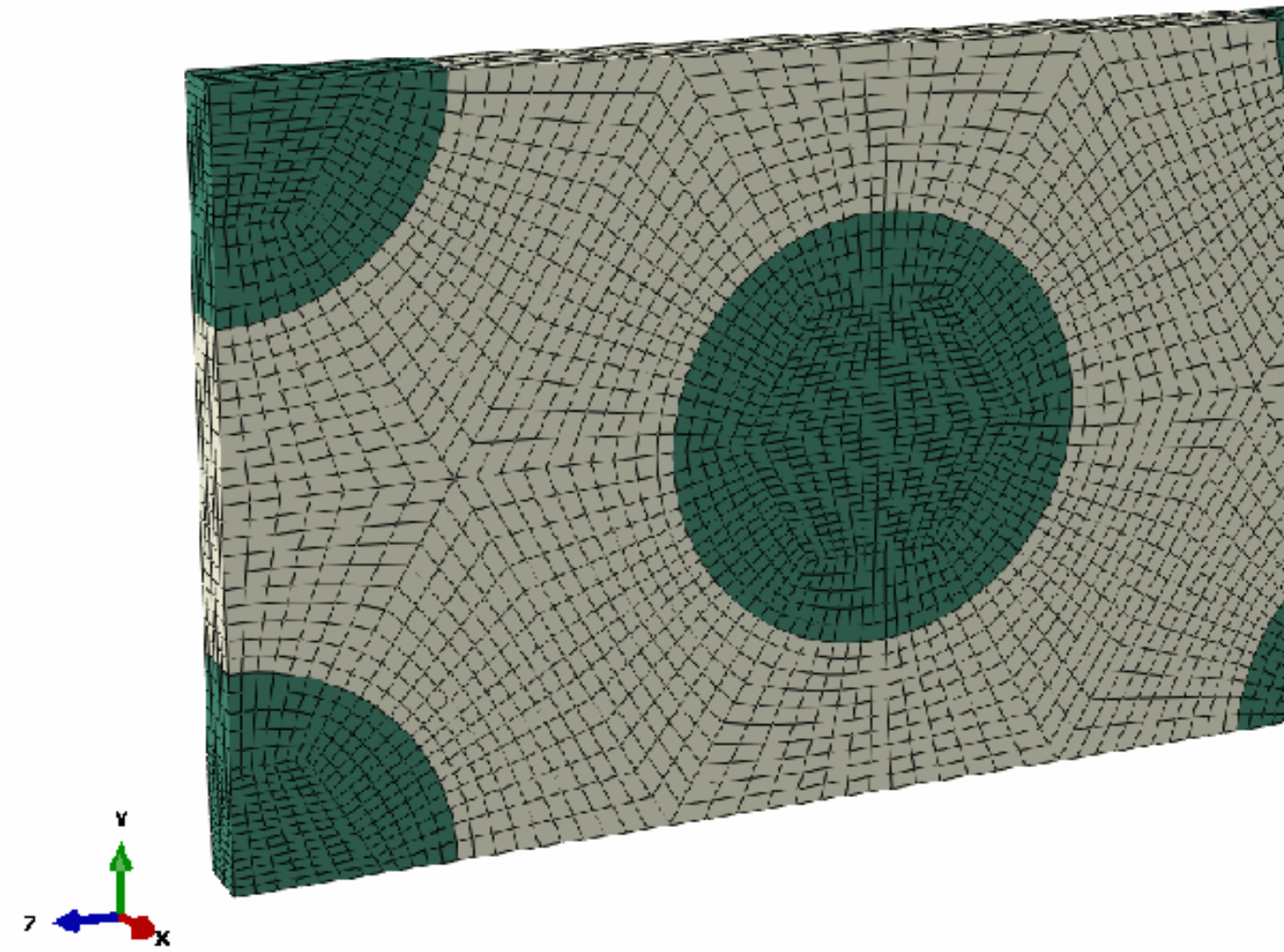


Figure 6: Effect of aspect ratio on macroscale thermal expansion coefficient predicted by the Mori-Tanaka homogenization method. Comparison is made between composites with different volume fractions.


Y
Z
X

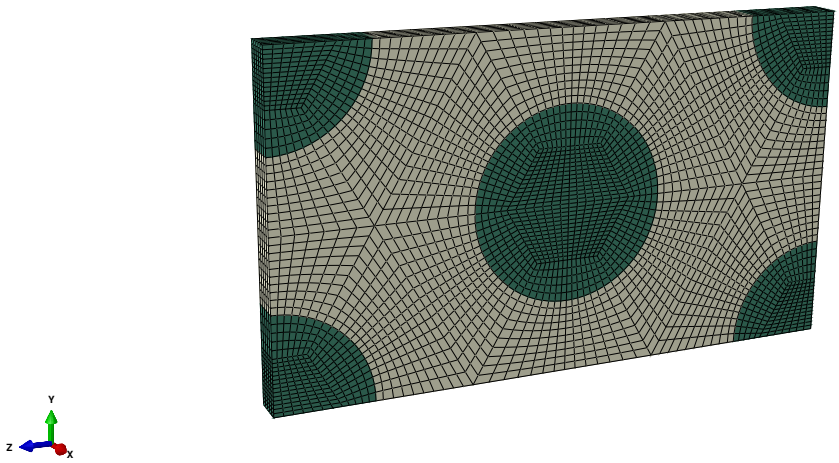


Figure 7: Hex-packed FE-RVE with cylindrical inclusions and 30% of fiber volume fraction.

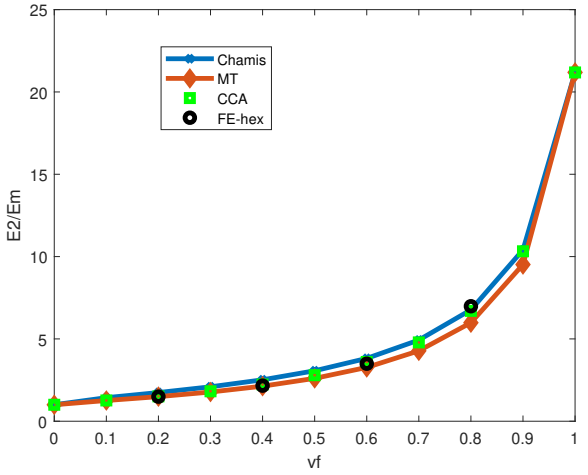


Figure 8: Comparison of homogenized elastic constant E_2 of epoxy-glass composite 7. The values are normalized using the Young's modulus of the matrix material.

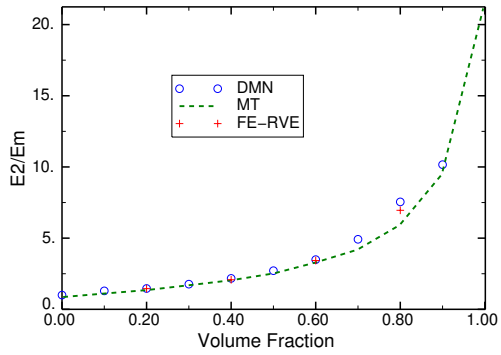


Figure 9: Comparison of homogenized elastic constant E_2 of epoxy-glass composite 7 between DMN, Mori-Tanaka, and FE-RVE predictions. The values are normalized using the Young's modulus of the matrix material.

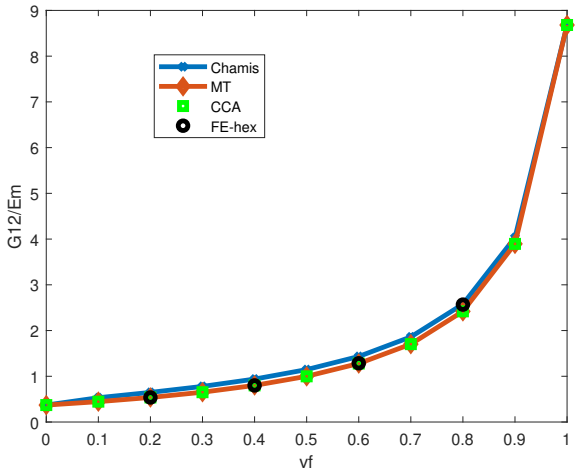


Figure 10: Comparison of homogenized elastic constant G_{12} of epoxy-glass composite 7. The values are normalized using the Young's modulus of the matrix material.

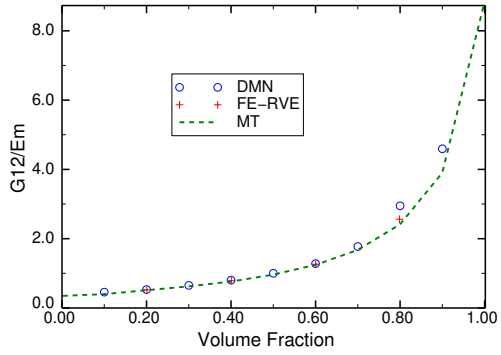


Figure 11: Comparison of homogenized elastic constant G_{12} of epoxy-glass composite 7 between DMN, Mori-Tanaka, and FE-RVE. The values are normalized using the Young's modulus of the matrix material.

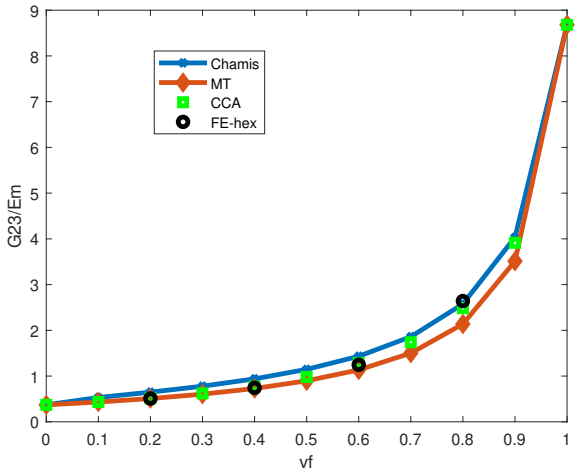


Figure 12: Comparison of homogenized elastic constant G_{23} of epoxy-glass composite 7. The values are normalized using the Young's modulus of the matrix material.

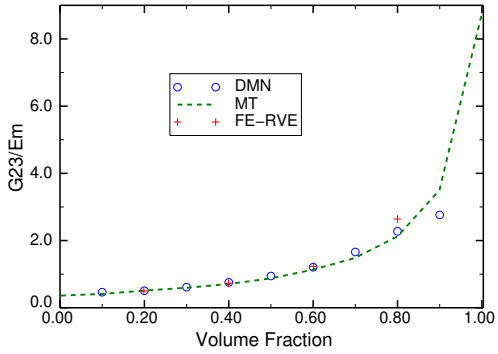


Figure 13: Comparison of homogenized elastic constant G_{23} of epoxy-glass composite 7 between DMN, Mori-Tanaka, and FE-RVE. The values are normalized using the Young's modulus of the matrix material.

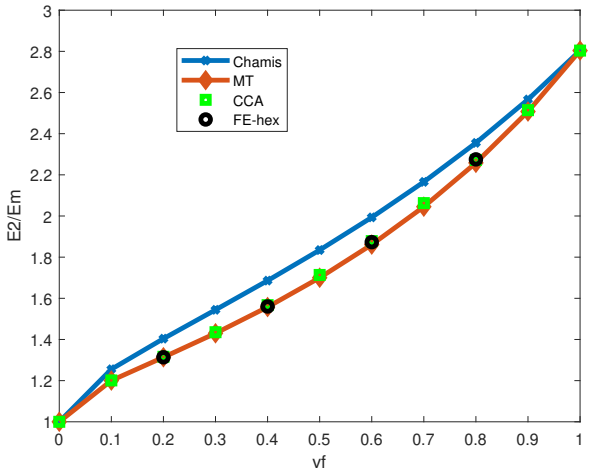


Figure 14: Comparison of homogenized elastic constant E_2 of epoxy-carbon composite 8. The values are normalized using the Young's modulus of the matrix material.

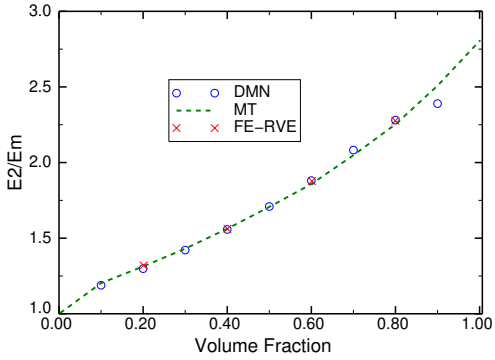


Figure 15: Comparison of homogenized elastic constant E_2 of epoxy-carbon composite 8 between DMN, Mori-Tanaka and, FE-RVE. The values are normalized using the Young's modulus of the matrix material.

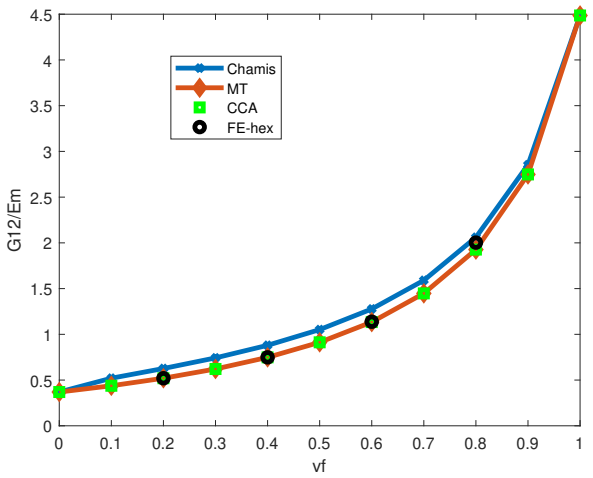


Figure 16: Comparison of homogenized elastic constant G_{12} of epoxy-carbon composite 8. The values are normalized using the Young's modulus of the matrix material.

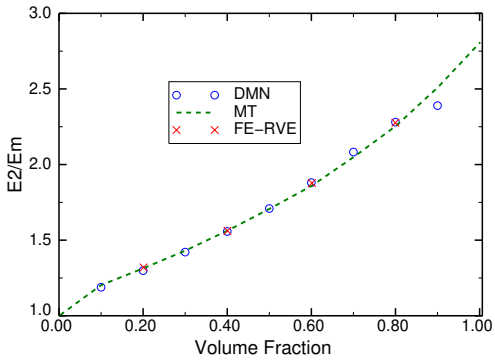


Figure 17: Comparison of homogenized elastic constant G_{12} of epoxy-carbon composite 8 between DMN, Mori-Tanaka, and FE-RVE. The values are normalized using the Young's modulus of the matrix material.

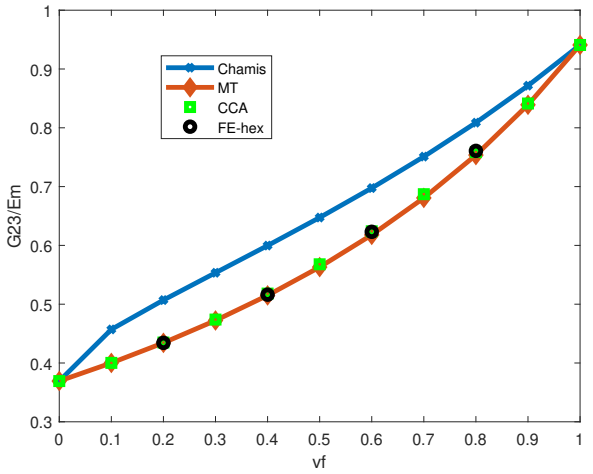


Figure 18: Comparison of homogenized elastic constant G_{23} of epoxy-carbon composite 8. The values are normalized using the Young's modulus of the matrix material.

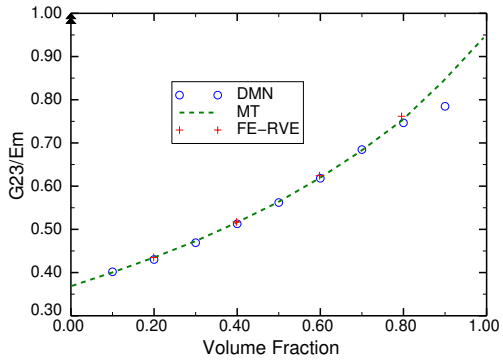


Figure 19: Comparison of homogenized elastic constant G_{23} of epoxy-carbon composite 8 between DMN, Mori-Tanaka, and FE-RVE. The values are normalized using the Young's modulus of the matrix material.

Multiscale linear viscoelastic materials

Products: Abaqus/Standard

References:

- *Mean-Field Homogenization*
- *Multiscale Material Modeling*

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D8R

Features tested

Linear elastic composites with viscoelasticity modeled with MFH and DMN methods.

Problem description

The verification tests consist of a set of single-element tests that use C3D8R elements under uniaxial tension loading conditions. Transient static responses are predicted.

Material:

Composite 1

Matrix properties:

E	70 GPa
ν	0.33

Prony series coefficients (N=2): $\overline{g}_1^P = 0.55$, $\overline{k}_1^P = 0$, $\tau_1 = 1.766$, $\overline{g}_2^P = 0.4$, $\overline{k}_2^P = 0$, $\tau_2 = 0.1536$

Inclusion properties:

E	300 GPa
ν	0.2
v_f	0.1
a_r	20.0

The inclusion shape is prolate, and the directions are aligned.

Composite 2

Matrix properties:

E	70 GPa
ν	0.33

Prony series coefficients (N=2): $\overline{g}_1^P = 0.55$, $\overline{k}_1^P = 0$, $\tau_1 = 1.766$, $\overline{g}_2^P = 0.4$, $\overline{k}_2^P = 0$, $\tau_2 = 0.1536$

Inclusion properties (engineering constants):

E_1	240GPa
E_2	15GPa
E_3	15GPa
ν_{12}	0.2
ν_{13}	0.2
ν_{23}	0.07
G_{12}	27GPa
G_{13}	27GPa
G_{23}	7GPa
v_f	0.1
a_r	20.0

The inclusion shape is prolate, and the directions are aligned.

Results and discussion

Transient static responses of the composites predicted by MFH and DMN methods are validated against FE-RVE results for the same microstructures.

The macroscale response of Composite 1 under uniaxial tension is plotted in *Figure 1*. The Mori-Tanaka method is used for the MFH analysis. Both the MFH and the DMN results compare well with the FE-RVE prediction.

The macroscale responses of Composite 2 under uniaxial tension in both the axial and transverse directions are plotted in *Figure 2*. The Mori-Tanaka method is used for the MFH analysis. Both the MFH and the DMN results compare well with the FE-RVE prediction.

Input files

multscale-prolate-vf10-visc.inp	One-element uniaxial tension test using Composite 1. Load is applied in the axial direction of the inclusion. Mori-Tanaka homogenization is used.
dmn-prolate-vf10-visc.inp	One-element uniaxial tension test using Composite 1. Load is applied in the axial direction of the inclusion. DMN is used.
rve-prolate-vf10-visc.inp	FE-RVE model of Composite 1 under uniaxial tension loading. Load is applied in the axial direction of the inclusion.
multscale-prolate-vf10-visc-EC.inp	One-element uniaxial tension test using Composite 2. Load is applied in the axial direction of the inclusion. Mori-Tanaka homogenization is used.
dmn-prolate-vf10-visc-EC.inp	One-element uniaxial tension test using Composite 2. Load is applied in the axial direction of the inclusion. DMN is used.

rve-prolate-vf10-visc-EC.inp	FE-RVE model of Composite 2 under uniaxial tension loading. Load is applied in the axial direction of the inclusion.
multscale-prolate-vf10-visc-EC-transverse.inp	One-element uniaxial tension test using Composite 2. Load is applied in the transverse direction of the inclusion. Mori-Tanaka homogenization is used.
dmn-prolate-vf10-visc-EC-transverse.inp	One-element uniaxial tension test using Composite 2. Load is applied in the transverse direction of the inclusion. DMN is used.
rve-prolate-vf10-visc-EC-transverse.inp	FE-RVE model of Composite 2 under uniaxial tension loading. Load is applied in the transverse direction of the inclusion.

Figures

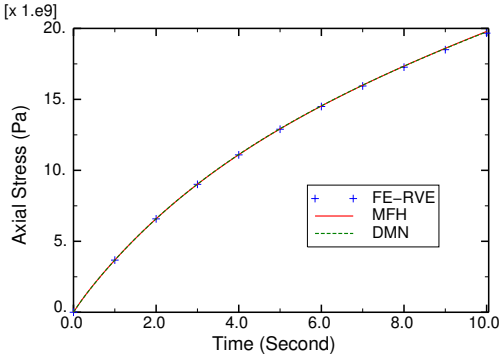


Figure 1: Prediction of the time response of the stress in a uniaxial test of a composite with viscoelasticity defined in the matrix material. Inclusion material is isotropic.

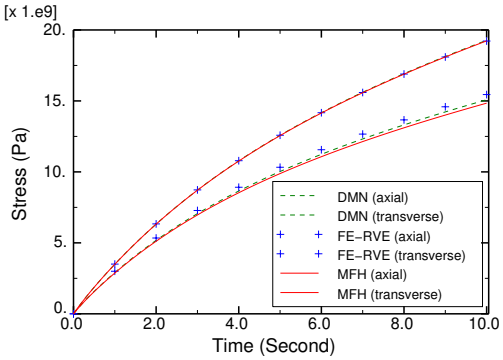


Figure 2: Prediction of the time response of the stress in a unitension test of a composite with viscoelasticity defined in the matrix material. Inclusion material is orthotropic. Load is applied in the axial and transverse directions separately.

Multiscale nonlinear materials

Products: Abaqus/Standard

References:

- *Mean-Field Homogenization*
- *Multiscale Material Modeling*

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D8R

Features tested

Nonlinear multiscale materials modeled with MFH and DMN methods.

Problem description

The verification tests consist of a set of single-element tests that use C3D8R elements under uniaxial tension or simple shear loading conditions. Hardening of the matrix material in Composite 1 and Composite 2 is defined with the following power law:

$$\overline{\sigma} = \sigma_Y + k\left(\overline{\epsilon}^{pl}\right)^n,$$

where $\overline{\sigma}$ is the yield stress and $\overline{\epsilon}^{pl}$ is the equivalent plastic strain. σ_Y is the yield stress at zero plastic strain. k and n are hardening parameters.

Hardening of the matrix material in Composite 3 is defined with the following exponential-linear law:

$$\overline{\sigma} = \sigma_Y + k_1\overline{\epsilon}^{pl} + k_2\left(1 - \exp\left(-m\overline{\epsilon}^{pl}\right)\right),$$

where k_1, k_2, and m are hardening parameters.

For both constituents in Composite 4, a power law is used to define hardening, and the rate-dependent yield behavior is defined by the Cowper-Symonds overstress power law, which has the form

$$\dot{\overline{\epsilon}}^{pl} = D\left(\frac{\overline{\sigma}}{\sigma^0} - 1\right)^m,$$

where $\sigma^0 = \sigma^0\left(\overline{\epsilon}^{pl}\right)$ is the static stress-strain behavior. The same nonlinear behavior is defined for the matrix in Composite 5.

Material:

Composite 1

Matrix properties:

E	75 GPa
ν	0.3

σ_Y	75 MPa
k	416 MPa
n	0.3895

The matrix is reinforced with spherical inclusion with the following properties:

E	400 GPa
ν	0.2
v_f	0.1
a_r	1.0

Composite 2a

Matrix properties:

E	72.5 GPa
ν	0.34
σ_Y	250 MPa
k	173 MPa
n	0.46

The matrix is reinforced by randomly distributed short fibers. The fiber properties are as follows:

E	455 GPa
ν	0.15
v_f	0.2
a_r	10.0

Composite 2b

Constituent properties are the same as Composite 2a. Short fibers are distributed axisymmetrically in the 1-2 plane.

Composite 3a

Matrix properties:

E	2.1 GPa
ν	0.3
σ_Y	29 MPa
k_1	132 MPa
k_2	32.7 MPa

m	319.4

The matrix is reinforced with short fibers that are aligned in the 1-direction. The fiber properties are as follows:

E	72 GPa
ν	0.22

Composite 3b

Constituent properties are the same as Composite 3a. Short fibers are distributed randomly in the 1-2 plane.

Composite 4

Matrix properties:

E	100 GPa
ν	0.3
σ_Y	100 MPa
k	5 GPa
n	1.0
D	$0.3 \times 10^{-3} s^{-1}$
m	1.0

The matrix is reinforced with spherical inclusion. The inclusion properties are as follows:

E	500 GPa
ν	0.3
σ_Y	500 MPa
k	10 GPa
n	0.8
D	$0.6 \times 10^{-3} s^{-1}$
m	2.5
v_f	0.1
a_r	1.0

Composite 5

Matrix properties are the same as Composite 4. The matrix is reinforced with spherical inclusions that are linear elastic:

E	500 GPa
ν	0.3

Results and discussion

Nonlinear macroscale properties predicted by MFH and DMN are validated.

The macroscale response of Composite 1 under uniaxial tension and simple shear loading is plotted in *Figure 1* and *Figure 2*. The Mori-Tanaka method is used for the MFH analysis. Both the MFH and the DMN results compare well with the FE-RVE prediction. The results from both the uniaxial tension and the simple shear tests show that the isotropization methods have little influence on the Mori-Tanaka prediction for this type of composite.

The macroscale response of Composite 2a under uniaxial tension loading is plotted in *Figure 3*. The Mori-Tanaka method is used for homogenization. 10×10 angle subdivisions are used to compute the numerical integration over orientation angles. The results are compared with those presented by *Doghri (2006)*.

The macroscale response of Composite 2b under uniaxial tension loading is plotted in *Figure 4*. DMN is used to predict the macroscale response. The results compared well with measured data presented in *Dunn (1997)*.

The macroscale response of Composite 3a under uniaxial tension loading is plotted in *Figure 5*. The matrix material is reinforced with short fibers that are aligned in the 1-direction. The DMN trained with hex-packed prolate inclusions (DMN_UD) and the DMN trained with distributed short fiber composites (DMN_CHOPPED) are used for prediction; the orientation tensor is specified as (1,0,0,0,0,0) when DMN_CHOPPED is used. The two DMN predictions are close. The DMN results are also compared with the FE-RVE results and the second-order MFH prediction presented in *Doghri (2011)*.

The macroscale response of Composite 3b under uniaxial tension loading is plotted in *Figure 6*. DMN results are compared with the FE-RVE results and the second-order MFH prediction presented in *Doghri (2011)*.

The macroscale response of Composite 4 under uniaxial tension loading is plotted in *Figure 7*. The Mori-Tanaka homogenization method is used. The results compare well with the FE-RVE prediction.

The macroscale response of Composite 5 under uniaxial tension loading is plotted in *Figure 8*. The Mori-Tanaka homogenization method is used for MFH. DMN predictions are more accurate than MFH when compared with FE-RVE results.

Input files

multscale-sphere-vf10-eiso.inp	One-element uniaxial tension test using Composite 1. Mori-Tanaka homogenization is used. The general method is used for isotropization, and the isotropized matrix modulus is applied only to the Eshelby tensor calculation.
multscale-sphere-vf10-piso.inp	One-element uniaxial tension test using Composite 1. Mori-Tanaka homogenization is used. The spectral method is used for isotropization, and the isotropized matrix modulus is applied to the calculation of Hill's tensor.
multscale-sphere-vf10-shear-eiso.inp	One-element simple shear test using Composite 1. Mori-Tanaka homogenization is used. The general method is used for isotropization, and the isotropized matrix modulus is applied only to the Eshelby tensor calculation.
multscale-sphere-vf10-shear-piso.inp	One-element simple shear test using Composite 1. Mori-Tanaka homogenization is used. The spectral method is used for isotropization, and the isotropized matrix modulus is applied to the calculation of Hill's tensor.
dmn-sphere-vf10-elasplas.inp	One-element uniaxial tension test using Composite 1. DMN is used.
dmn-sphere-vf10-elasplas-shear.inp	One-element simple shear test using Composite 1. DMN is used.
rve-sphere-vf10-elasplas.inp	FE-RVE model of Composite 1 under uniaxial tension loading.
rve-sphere-vf10-elasplas-shear.inp	FE-RVE model of Composite 1 under simple shear loading.
multscale-prolate-random3d.inp	One-element uniaxial tension test using Composite 2a.

dmn-prolate-random2d.inp	One-element uniaxial tension test using Composite 2b. DMN trained with distributed short fiber reinforced composites is used.
dmn_prolate_aligned.inp	One-element uniaxial tension test using Composite 3a. DMN trained with hex-packed prolate-shaped fiber reinforced composites is used.
dmn_chopped_aligned.inp	One-element uniaxial tension test using Composite 3a. DMN trained with distributed short fibers reinforced composites is used.
dmn_chopped_random2d.inp	One-element uniaxial tension test using Composite 3b. DMN trained with distributed short fiber reinforced composites is used.
multscale-sphere-visplas.inp	One-element uniaxial tension test using Composite 4.
dmn-sphere-visplas-elasfiber.inp	One-element uniaxial tension test using Composite 5.
rve-sphere-visplas.inp	FE-RVE model of Composite 4 under uniaxial tension loading.
power-uhard.f	User subroutine to define the power law hardening and power law rate dependency for Composite 3.

References

- Doghri, I., and A. Ouaar, "Homogenization of Two-Phase Elasto-Plastic Composite Materials and Structures: Study of Tangent Operators, Cyclic Plasticity and Numerical Algorithms," *International Journal of Solids and Structures*, vol. 40, pp. 1681–1712, 2003.
- Doghri, I., and L. Tinel, "Micromechanics of Inelastic Composites with Misaligned Inclusions: Numerical Treatment of Orientation," *Computational Methods in Applied Mechanics and Engineering*, vol. 195, pp. 1387–1406, 2006.
- Dunn, M.L., , and H. Ledbetter, "Elastic-Plastic Behavior of Textured Short-Fiber Composites," *Acta Materialia*, vol. 45, 1997.
- Pierard, O., and I. Doghri, "An Enhanced Affine Formulation and the Corresponding Numerical Algorithms for the Mean-Field Homogenization of Elasto-Viscoplastic Composites," *International Journal of Plasticity*, vol. 22, pp. 131–157, 2006.
- Doghri, I., , and L. Brassart, "A Second-Moment Incremental Formulation for the Mean-Field Homogenization of Elasto-Plastic Composites," *International Journal of Plasticity*, vol. 27, 2011.

Figures

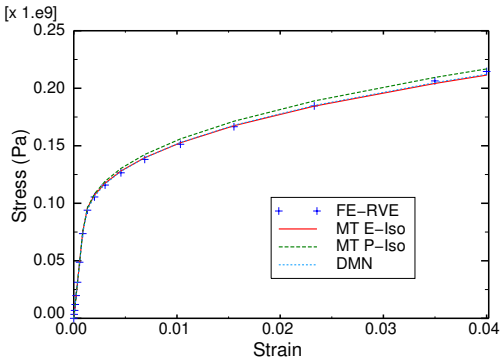


Figure 1: Prediction of the macroscale stress in a uniaxial test of a composite with elastic-plastic matrix reinforced with elastic spherical inclusions.

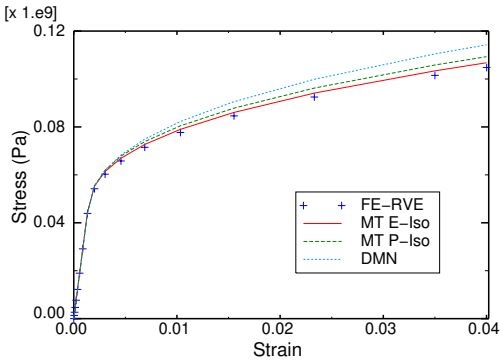


Figure 2: Prediction of the macroscale stress in a simple shear test of a composite with elastic-plastic matrix reinforced with elastic spherical inclusions.

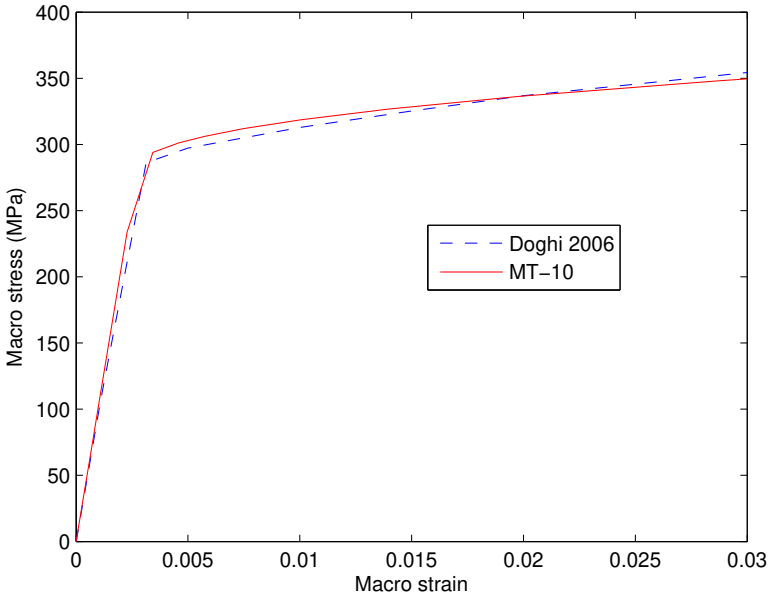


Figure 3: Prediction of the macroscale stress in a uniaxial test of an elastic-plastic matrix reinforced with randomly distributed fibers.

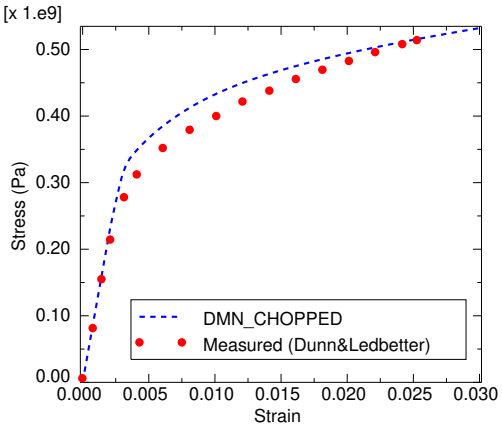


Figure 4: Prediction of the macroscale stress in a uniaxial test of an elastic-plastic matrix reinforced with axisymmetrically distributed fibers.

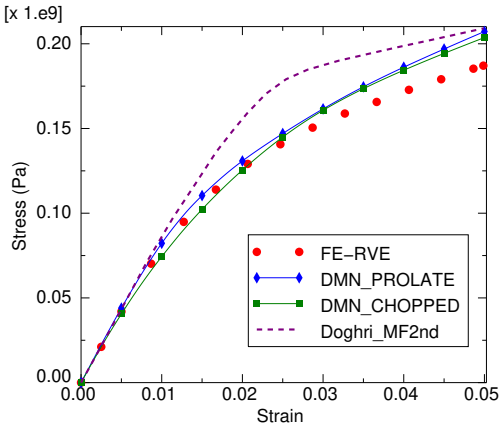


Figure 5: Prediction of the macroscale stress in a uniaxial test of an elastic-plastic matrix reinforced with aligned short fibers.

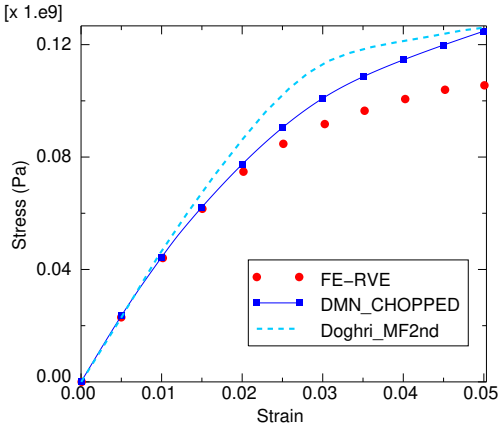


Figure 6: Prediction of the macroscale stress in a uniaxial test of an elastic-plastic matrix reinforced with short fibers. Fibers are randomly distributed in the X-Y plane.

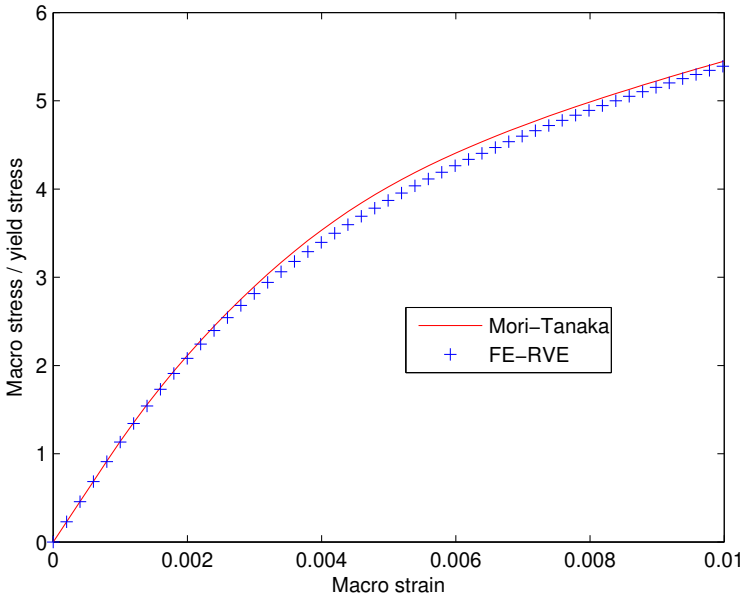


Figure 7: Normalized macroscale stress of composite 4 in a uniaxial tension test with a strain rate of $10^{-3}s^{-1}$.

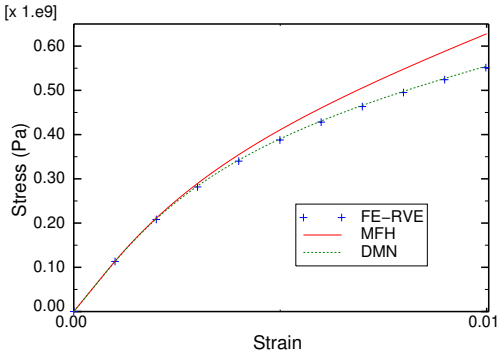


Figure 8: Macroscale stress of composite 5 in a uniaxial tension test with a strain rate of $10^{-3}s^{-1}$.

Multiscale materials with cure modeling

Products: Abaqus/Standard

References:

- *Mean-Field Homogenization*
- *Multiscale Material Modeling*

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D8RT CAX4RT CPE4RT CPE4RHT
DC2D4 DC3D8 DCAX4

Features tested

Multiscale materials with cure modeling.

Problem description

These tests verify the multiscale material model when cure properties are specified in the matrix material. Material properties specified in *Cure modeling in Abaqus/Standard* are used in these tests for the matrix material. The fiber volume fraction is set to a very small value; therefore, the results are identical to the test results in *Cure modeling in Abaqus/Standard*.

Results and discussion

The results of the multiscale model tests are identical with the results of the cure modeling tests in *Implementation using dedicated cure modeling options* and *Bilayer test*.

Input files

mfh_cureModel_Grindling_iso_cax4rt.inp	Multiscale material with Grindling reaction kinetics model, isotropic cure shrinkage strain, and CAX4RT element.
mfh_cureModel_Grindling_iso_cpe4rt.inp	Multiscale material with Grindling reaction kinetics model, isotropic cure shrinkage strain, and CPE4RT element.
mfh_cureModel_Grindling_iso_c3d8rt.inp	Multiscale material with Grindling reaction kinetics model, isotropic cure shrinkage strain, and C3D8RT element.
mfh_cureModel_Grindling_iso_max_conversion_cpe4rt.inp	Multiscale material with Grindling reaction kinetics model, isotropic cure shrinkage strain, CPE4RT element, and maximum conversion.
mfh_cureModel_Grindling_dc2d4_HT.inp	Multiscale material with Grindling reaction kinetics model, transient heat transfer analysis, and DC2D4 element.
mfh_cureModel_Grindling_max_conversion_dc2d4_HT.inp	Multiscale material with Grindling reaction kinetics model, transient heat transfer analysis, DC2D4 element, and maximum conversion.
mfh_cureModel_Kamal_iso_cpe4rt.inp	Multiscale material with Kamal reaction kinetics model, isotropic cure shrinkage strain, and CPE4RT element.

mfh_cureModel_Kamal_vol_cpe4rt.inp	Multiscale material with Kamal reaction kinetics model, volumetric cure shrinkage strain, and CPE4RT element.
mfh_cureModel_Kamal_ortho_cpe4rt.inp	Multiscale material with Kamal reaction kinetics model, orthotropic cure shrinkage strain, and CPE4RT element.
mfh_cureModel_Kamal_iso_c3d8rt.inp	Multiscale material with Kamal reaction kinetics model, isotropic cure shrinkage strain, and C3D8RT element.
mfh_cureModel_Kamal_iso_max_conversion_cpe4rt.inp	Multiscale material with Kamal reaction kinetics model, isotropic cure shrinkage strain, CPE4RT element, and maximum conversion.
mfh_cureModel_Kamal_dc2d4_HT.inp	Multiscale material with Kamal reaction kinetics model, transient heat transfer analysis, and DC2D4 element.
mfh_cureModel_Kamal_dc3d8_HT.inp	Multiscale material with Kamal reaction kinetics model, transient heat transfer analysis, and DC3D8 element.
mfh_cureModel_Kamal_dcax4_HT.inp	Multiscale material with Kamal reaction kinetics model, transient heat transfer analysis, and DCAX4 element.
mfh_cureModel_Kamal_max_conversion_dc2d4_HT.inp	Multiscale material with Kamal reaction kinetics model, transient heat transfer analysis, DC2D4 element, and maximum conversion specified.
mfh_cureModel_tabular_iso_cpe4rt.inp	Multiscale material with tabular reaction kinetics model, isotropic cure shrinkage strain, and CPE4RT element.
mfh_cureModel_tabular_dc2d4_HT.inp	Multiscale material with tabular reaction kinetics model, transient heat transfer analysis, and DC2D4 element.
mfh_cureModel_Kamal_bilayer.inp	Bilayer test, multiscale material with Kamal reaction kinetics, isotropic cure shrinkage strain, CPE4RT and CPE4RHT elements.
mfh_ucure_Kamal_dc2d4_HT.inp	Multiscale material with Kamal reaction kinetics model defined in user subroutine *UCURE* and thermal properties defined in user subroutine *UMATHT*, transient heat transfer analysis, and DC2D4 element.
ucure_Kamal_umatht.f	User subroutines *UCURE* and *UMATHT* used in `mfh_ucure_Kamal_dc2d4_HT.inp`.

Cure modeling in Abaqus/Standard

Products: Abaqus/Standard

References:

- *Curing Processes in Polymers*
- *Modeling the Cure Process in Thermosetting Polymers*

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D8RT CAX4RT CPE4RT CPE4RHT
DC2D4 DC2D4E DC3D8 DC3D8E DCAX4 DCAX4E

Problem description

These tests are used to verify the cure modeling capabilities in Abaqus/Standard. The tests verify the correctness of the degree of cure, the cure shrinkage strain, and the amount of heat generation computed by the model. The tests verify the cure modeling capabilities for both the implementation using special-purpose built-in material options (see *Modeling the Cure Process in Thermosetting Polymers*) and the implementation using dedicated cure modeling options and output variables (see *Curing Processes in Polymers*).

Most of the tests consist of one element. In those tests, no stresses or thermal loads are applied, and sufficient boundary conditions are prescribed to prevent rigid body motion. In all cases, an initial temperature of 75 is prescribed. To begin the curing process, initial conditions are used to assign small initial values of degree of cure equal to 0.001.

In addition to the one-element tests, this verification includes a curing model of a bilayer strip (see *Figure 1*). The model consists of a steel strip coated with an adhesive. The properties of the adhesive material are identical to those used in the one-element tests. The steel is modeled as a linear elastic material with properties given in *Table 11*. The model setup is representative of the Kappa Gauge method used to study deformation developed during curing processes due to thermal expansion coefficients mismatch and cure shrinkage (see *Measurement of web curl[Lindeman, 2021]*). For verification purposes, we use the model to verify the correctness of the results at the location indicated in *Figure 1*.

Material: The material parameters used are given below.

Elasticity

Table 1: Linear elasticity.

E	ν	Temperature	Field variable
0.003	0.45	20.0	0.0
19.2	0.45	200.0	1.0

Viscoelasticity

Table 2: Prony series coefficients.

g_1	k_1	τ_1
0.4	0.0	2×10^{-7}
0.3	0.0	4.0
0.2	0.0	5×10^{8}

Table 3: Tabular definition of the shift function.

Logarithm of the shift function	Temperature	Field variable
-3.5	20.0	0.0
-3.0	200.0	0.0
0.5	20.0	1.0
-16.0	200.0	1.0

Thermal properties

Table 4: Conductivity and specific heat.

k	c
3.2×10^{-4}	1.86×10^{6}

Reaction kinetics

Table 5: Grindling model.

A_1	E_1/R	n_1	A_2	E_2/R	n_2	m	c_1	c_2	$K_D^{\theta=\theta_g}$	$\Delta\theta_g$
2.07×10^{6}	6.67×10^{4}	2.0	1.5×10^{5}	6.67×10^{3}	1.5	0.25	5.0	0.0	1.0×10^{3}	50.0

Table 6: Kamal model.

Z_1	E_1	m_1	n_1	b_1
3.5×10^{14}	10^{5}	1.05	1.3	0.0

The values for the tabular model were generated from the Kamal model described in the table above.

Shrinkage coefficients

Table 7: Volumetric and isotropic shrinkage coefficients.

Isotropic shrinkage coefficient, γ	Volumetric shrinkage coefficient, γ_{vol}
0.01433	0.043

Table 8: Orthotropic shrinkage coefficients.

γ_{11}	γ_{22}	γ_{33}
0.01433	0.01	0.007

Table 9: Anisotropic shrinkage coefficients.

γ_{11}	γ_{22}	γ_{33}	γ_{12}	γ_{13}	γ_{23}
0.01433	0.01	0.007	0.002	0.001	0.0015

Volumetric heat generation parameters

Table 10: Specific heat of reaction and initial density.

Q	ρ_0
2.5×10^7	1.25×10^{-9}

Material properties for steel in bilayer strip model

Table 11: Steel material properties.

Young's modulus	Poisson's ratio	Thermal conductivity	Specific heat	Coefficient of thermal expansion
193000.0	0.29	0.0162	500000.0	1.73×10^{-5}

Results and discussion

The degree of cure results agree well with those obtained using a semianalytical method. The cure shrinkage strain and the volumetric heat generated as the curing reaction progresses agree well with the results obtained analytically. For the bilayer strip model, the equivalent Mises stress and the degree of cure at the location shown in *Figure 1* are plotted in *Figure 2* and *Figure 3*, respectively.

Input files

Implementation using special-purpose built-in material options

cure_Kamal_iso_cax4rt.inp	Kamal reaction kinetics model, isotropic cure shrinkage strain, and CAX4RT element.
cure_Kamal_iso_cpe4rt.inp	Kamal reaction kinetics model, isotropic cure shrinkage strain, and CPE4RT element.
cure_tabular_iso_cpe4rt.inp	Tabular reaction kinetics model, isotropic cure shrinkage strain, and CPE4RT element.
cure_Kamal_vol_cpe4rt.inp	Kamal reaction kinetics model, volumetric cure shrinkage strain, and CPE4RT element.
cure_Kamal_ortho_cpe4rt.inp	Kamal reaction kinetics model, orthotropic cure shrinkage strain, and CPE4RT element.
cure_Kamal_aniso_cpe4rt.inp	Kamal reaction kinetics model, anisotropic cure shrinkage strain, and CPE4RT element.
cure_Kamal_iso_c3d8rt.inp	Kamal reaction kinetics model, isotropic cure shrinkage strain, and C3D8RT element.
cure_Kamal_iso_max_conversion_cpe4rt.inp	Kamal reaction kinetics model, isotropic cure shrinkage strain, CPE4RT element, and maximum conversion specified.

Input files containing definitions or data to be included in the input files listed above

ABQ_Cure_types.inp	Types of property tables and parameter tables used by the special-purpose built-in material for simulation of the cure process.

Implementation using dedicated cure modeling options

cureModel_Kamal_iso_cax4rt.inp	Kamal reaction kinetics model, isotropic cure shrinkage strain, and CAX4RT element.
cureModel_Grindling_iso_cax4rt.inp	Grindling reaction kinetics model, isotropic cure shrinkage strain, and CAX4RT element.
cureModel_Kamal_iso_cpe4rt.inp	Kamal reaction kinetics model, isotropic cure shrinkage strain, and CPE4RT element.
cureModel_tabular_iso_cpe4rt.inp	Tabular reaction kinetics model, isotropic cure shrinkage strain, and CPE4RT element.
cureModel_Grindling_iso_cpe4rt.inp	Grindling reaction kinetics model, isotropic cure shrinkage strain, and CPE4RT element.
cureModel_Kamal_vol_cpe4rt.inp	Kamal reaction kinetics model, volumetric cure shrinkage strain, and CPE4RT element.
cure_Kamal_ortho_cpe4rt.inp	Kamal reaction kinetics model, orthotropic cure shrinkage strain, and CPE4RT element.
cureModel_Kamal_aniso_cpe4rt.inp	Kamal reaction kinetics model, anisotropic cure shrinkage strain, and CPE4RT element.
cureModel_Kamal_iso_c3d8rt.inp	Kamal reaction kinetics model, isotropic cure shrinkage strain, and C3D8RT element.
cureModel_Grindling_iso_c3d8rt.inp	Grindling reaction kinetics model, isotropic cure shrinkage strain, and C3D8RT element.

cureModel_Kamal_iso_max_conversion_cpe4rt.inp	Kamal reaction kinetics model, isotropic cure shrinkage strain, CPE4RT element, and maximum conversion specified.
cureModel_Grindling_iso_max_conversion_cpe4rt.inp	Grindling reaction kinetics model, isotropic cure shrinkage strain, CPE4RT element, and maximum conversion specified.
cureModel_Kamal_iso_t2d2t.inp	Kamal reaction kinetics model, isotropic cure shrinkage strain, and T2D2T element.
cureModel_Grindling_iso_t2d2t.inp	Grindling reaction kinetics model, isotropic cure shrinkage strain, and T2D2T element.
cureModel_Kamal_dc2d4_HT.inp	Kamal reaction kinetics model, transient heat transfer analysis, and DC2D4 element.
cureModel_Grindling_dc2d4_HT.inp	Grindling reaction kinetics model, transient heat transfer analysis, and DC2D4 element.
cureModel_tabular_dc2d4_HT.inp	Tabular reaction kinetics model, transient heat transfer analysis, and DC2D4 element.
cureModel_Kamal_dc3d8_HT.inp	Kamal reaction kinetics model, transient heat transfer analysis, and DC3D8 element.
cureModel_Kamal_dcax4_HT.inp	Kamal reaction kinetics model, transient heat transfer analysis, and DCAX4 element.
cureModel_Kamal_max_conversion_dc2d4_HT.inp	Kamal reaction kinetics model, transient heat transfer analysis, DC2D4 element, and maximum conversion specified.
cureModel_Grindling_max_conversion_dc2d4_HT.inp	Grindling reaction kinetics model, transient heat transfer analysis, DC2D4 element, and maximum conversion specified.
cureModel_Kamal_dc2d4e_CTE.inp	Kamal reaction kinetics model, transient coupled thermal-electrical analysis, and DC2D4E element.
cureModel_Kamal_dc3d8e_CTE.inp	Kamal reaction kinetics model, transient coupled thermal-electrical analysis, and DC3D8E element.
cureModel_Grindling_dc3d8e_CTE.inp	Grindling reaction kinetics model, transient coupled thermal-electrical analysis, and DC3D8E element.
cureModel_Kamal_dcax4e_CTE.inp	Kamal reaction kinetics model, transient coupled thermal-electrical analysis, and DCAX4E element.
cureModel_Kamal_max_conversion_dc2d4e_CTE.inp	Kamal reaction kinetics model, transient coupled thermal-electrical analysis, DC2D4E element, and maximum conversion specified.
cureModel_tabular_dc2d4e_CTE.inp	Tabular reaction kinetics model, transient coupled thermal-electrical analysis, and DC2D4E element.
cureModel_Kamal_iso_fv_c3d8rt.inp	Kamal reaction kinetics model with parameters dependent on field variables, isotropic cure shrinkage strain, and C3D8RT element.

Bilayer test

cureModel_Kamal_bilayer.inp	Bilayer test, Kamal reaction kinetics, isotropic cure shrinkage strain, CPE4RT and CPE4RHT elements.

References

- Lindeman, D., S. Carpenter, and C. Li, Residual Stress Development During Curing of Structural Adhesives: Experimental Characterization and Modeling, *44th Meeting of the Adhesion Society*, 2021.

Figures

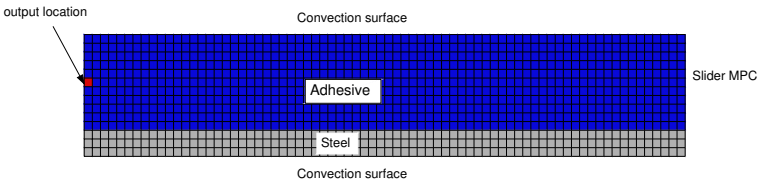


Figure 1: Bilayer strip model.

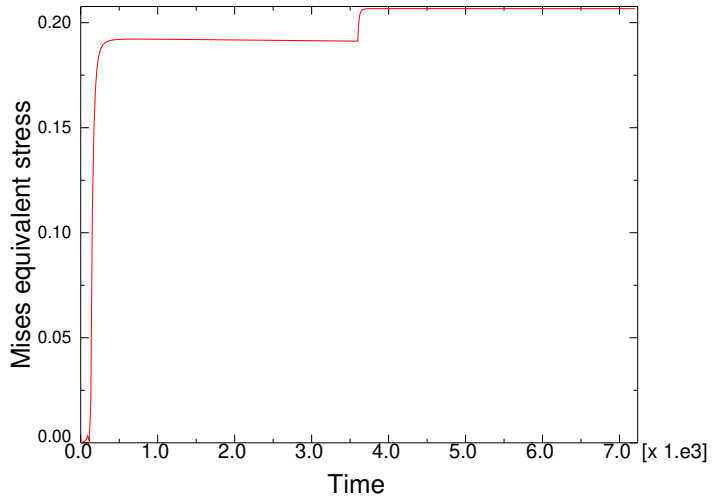


Figure 2: Mises equivalent stress vs. time.

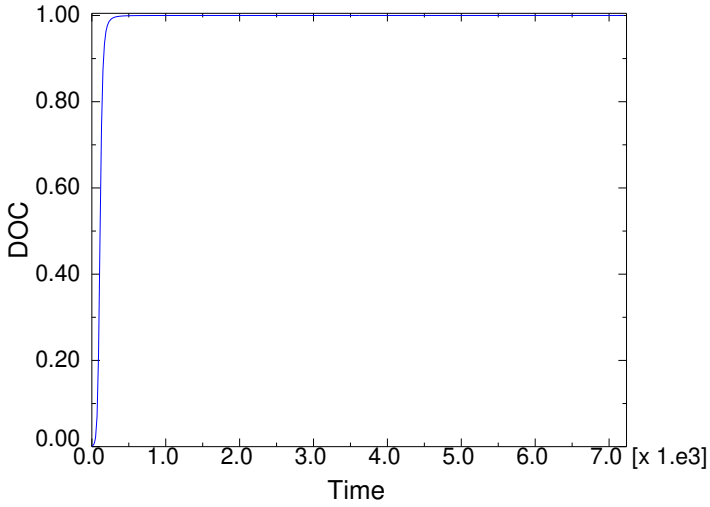


Figure 3: Degree of cure vs. time.

Plasticity corrections

Products: Abaqus/Standard

References:

- *Plasticity Corrections*

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D8 CPE4
S4
T3D2

Problem description

The tests are used to verify the correctness of the equivalent stresses and strains computed using the Neuber and Glinka rules in Abaqus/Standard.

All tests consist of four elements subjected to uniaxial tension. The same isotropic linearly elastic material definition is used in all cases, and three different methods are used to specify the plastic material properties used for the evaluation of the plasticity corrections: Ramberg-Osgood, tabular, and Mises yield surface. When the Ramberg-Osgood or the tabular method is used, Abaqus evaluates plasticity corrections in both general and static perturbation procedures. However, if an elastic-plastic material with Mises yield surface is used, the plasticity corrections are evaluated only in static perturbation procedures.

Material:
Elasticity

Young's modulus

$E = 3.0 \times 10^6$

Poisson's ratio

$\nu = 0.3$

Plasticity

Hardening

Yield stress	Plastic strain
10^6	0.0
2×10^6	1.0

Plasticity corrections

Ramberg-Osgood model

K	10^6
n	3.0

Tabular form

Yield stress	Plastic strain
10^6	0.0
2×10^6	1.0

Results and discussion

The numerically computed values of Neuber and Glinka equivalent stresses and strains agree well with the results obtained analytically.

Input files

pla_corr_ramberg_osgood.inp	C3D8, CPE4, S4, T3D2 elements; Ramberg-Osgood model; static procedure.
pla_corr_tabular.inp	C3D8, CPE4, S4, T3D2 elements; tabular form; static procedure.
pla_corr_ramberg_osgood_pert.inp	C3D8, CPE4, S4, T3D2 elements; Ramberg-Osgood model; static perturbation procedure.
pla_corr_tabular_pert.inp	C3D8, CPE4, S4, T3D2 elements; tabular form; static perturbation procedure.
pla_corr_pla_mises_pert.inp	C3D8, CPE4, S4, T3D2 elements; Mises yield surface; static perturbation procedure.
pla_corr_ramberg_osgood_2steps.inp	C3D8, CPE4, S4, T3D2 elements; Ramberg-Osgood model; static procedure followed by static perturbation procedure.
pla_corr_tabular_2steps.inp	C3D8, CPE4, S4, T3D2 elements; tabular form; static procedure followed by static perturbation procedure.
pla_corr_pla_mises_2steps.inp	C3D8, CPE4, S4, T3D2 elements; Mises yield surface; static procedure followed by static perturbation procedure.

Electrical and mechanical modeling of cardiac tissue

Products: Abaqus/Standard Abaqus/Explicit

References:

- *Modeling the Electrical Response of Cardiac Tissue*
- *Modeling the Mechanical Response of Cardiac Tissue*

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Electrical conduction in the absence of diffusion

Elements tested

DC3D8

Features tested

Electrical conduction in cardiac tissue in the absence of diffusion.

Problem description

The tests verify the electrical conduction in cardiac tissue in the absence of diffusion. In this context, the electric potential, ϕ, and recovery variable, r, are the solution of

$$\begin{cases} \dot{\phi} = \frac{\phi_N}{\tau} c\varphi(\phi)[1-\varphi(\phi)]\left[\varphi(\phi)-\alpha\right] - \frac{\phi_N}{\tau} r\varphi(\phi) \\ \dot{r} = \frac{1}{\tau}\left[\gamma + r\frac{\mu_1}{\mu_2+\varphi(\phi)}\right]\left[-r - c\varphi(\phi)\left[\varphi(\phi)-b-1\right]\right] \end{cases} \quad \text{and} \quad \begin{cases} \phi(t=0) = \phi_0 \\ r(t=0) = 0 \end{cases}$$

with $\varphi\left(\phi\right) = \frac{\phi-\phi_s}{\phi_N}$, ϕ_s = –80 mV, ϕ_N = 100 mV, and τ = 12.9 ms. This is a system of two coupled and nonlinear first-order ordinary differential equations that can easily be solved numerically.

The special-purpose framework implemented in Abaqus/Standard leverages the mathematical analogy between heat and electrical conduction. In this context, we interpret thermal quantities as their electrical counterparts and run the electrical analysis of cardiac tissue as the heat transfer analysis of an analogous thermally conductive material. Therefore, the tests use single DC3D8 elements subjected to the initial condition ϕ_0 = –50 mV and no other boundary conditions or loads. The simulation consists of a transient heat transfer analysis, which has a duration of 500 ms. A parameter table of type *"ABQ_CARDIAC_ELEC_ACTIVATION"* is specified with ϕ_{act} = 15 mV to record the activation time, t_{act}, in SD

Table 2 lists the material properties for materials with high or low refractoriness. Because there is no diffusion, the value of the conductivity does not influence the solution. However, we specify the diffusion alongside the specific heat and density because Abaqus/Standard requires these inputs, as shown in *Table 1*.

Material: The material parameters used in the tests are given below.

Table 1: Thermal conductivity, specific heat, and density of the analogous thermal material.

Thermal conductivity	1.0
Specific heat	1.0
Density	1.0

Table 2: Cardiac tissue properties.

Oscillation threshold α	0.05
Refractoriness γ	0.1 (high) or 0.001 (low)
Scaling parameter b	0.15
Scaling parameter c	8.0
Shape parameter μ_1	0.2
Shape parameter μ_2	0.3

(The units are not important.)

Results and discussion

The electric potential, ϕ, and recovery variable, r, obtained using the special-purpose material in Abaqus/Standard agree well with the corresponding results obtained using an independent numerical method. *Figure 1* and *Figure 2* show all the results.

Table 3 lists the activation times saved in SDV2. They are consistent with the results shown in *Figure 1*.

Input files

cardiac_electrical_high_refractoriness.inp	Electrical conduction in a material with high refractoriness in the absence of diffusion.
cardiac_electrical_low_refractoriness.inp	Electrical conduction in a material with low refractoriness in the absence of diffusion.

Tables

Table 3: Activation times.

High refractoriness	13.5 ms
Low refractoriness	11 ms

Figures

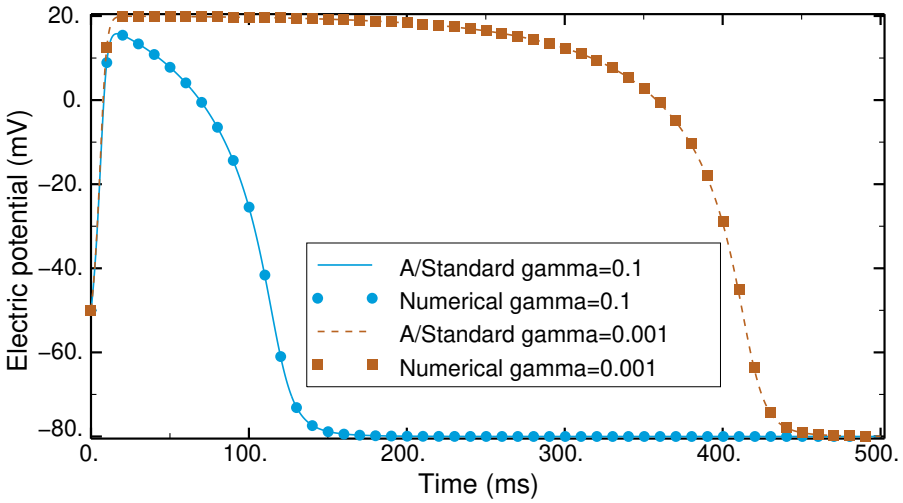


Figure 1: Electric potential vs. time.

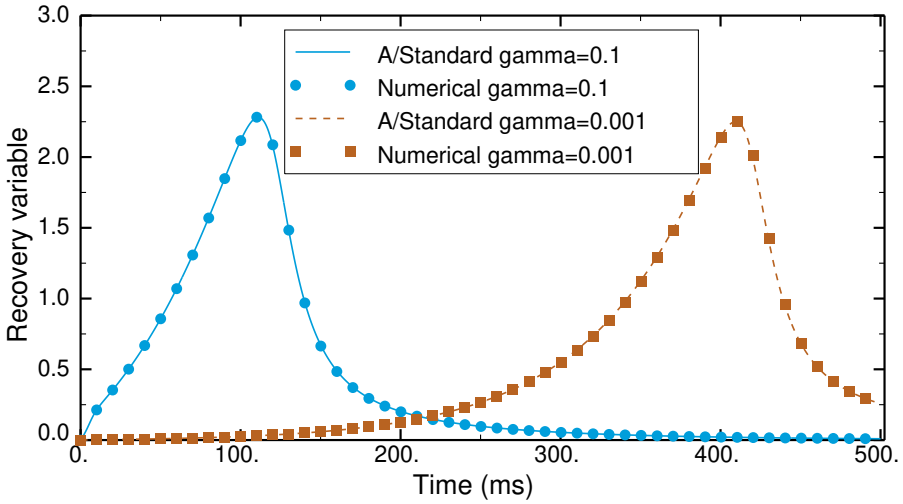


Figure 2: Recovery variable vs. time.

Electric cardiac response to a plane wave action potential

Elements tested

DC3D4

Features tested

Electrical cardiac response to a plane wave action potential and subsequent repolarization.

Problem description

A rectangular cuboid domain is utilized to verify the electrical analysis of cardiac tissue. A plane wave action potential is generated, traverses the simulation domain, and the cardiac tissue subsequently repolarizes to a resting potential. The problem is documented in Section 4.1 of *Göktepe et. al (2010)*.

The model consists of a rectangular cuboid domain with dimensions of 100 mm in the x- and y-directions and 1 mm in the z-direction. The mesh contains 118,595 linear DC3D4 elements. The simulation consists of a transient heat transfer analysis, which has a duration of 500 ms.

Material properties for the analogous thermally conductive material were taken from *Göktepe et. al (2010)* and are listed in *Table 4* and *Table 5*. The initial condition for SDV1 ($\gamma = 0.002$) overrides the value specified in the parameter table of type *"ABQ_CARDIAC_ELEC_PARAMETERS_AP"* ($\gamma = 0.001$).

The initial electrical potential throughout the domain is set to the resting potential of −80 mV. To create a plane wave action potential, the nodes on the left side of the domain are set to an initial electric potential of −40 mV.

Material: The material parameters used in the tests are given below.

Table 4: Thermal conductivity, specific heat, and density of the analogous thermal material.

Thermal conductivity	0.2
Specific heat	1.0
Density	1.0

Table 5: Cardiac tissue properties.

Oscillation threshold α	0.01
Refractoriness γ	0.002
Scaling parameter b	0.15
Scaling parameter c	8.0
Shape parameter μ_1	0.2
Shape parameter μ_2	0.3

(The units are not important.)

Results and discussion

Contour plots of the electric potential at 125 ms and 500 ms, as shown in *Figure 3* and *Figure 4*, respectively, show good agreement with those reported by *Göktepe et. al (2010)*.

Input files

cardiac_electrical_diffusion_recovery.inp	Electrical cardiac response to a plane wave action potential and subsequent repolarization.

References

- Göktepe , S., J. Wong, and E. Kuhl, “Atrial and Ventricular Fibrillation: Computational Simulation of Spiral Waves in Cardiac Tissue,” *Archive of Applied Mechanics*, vol. 80, pp. 569–580, 2010.

Figures

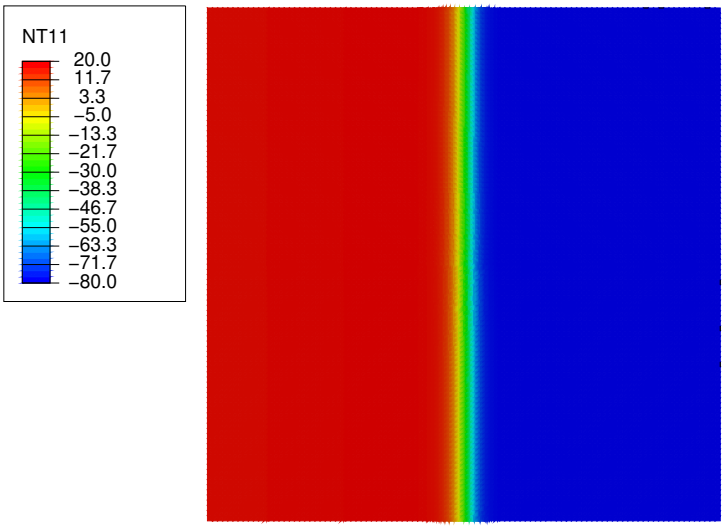


Figure 3: Electric potential at t = 125 ms.

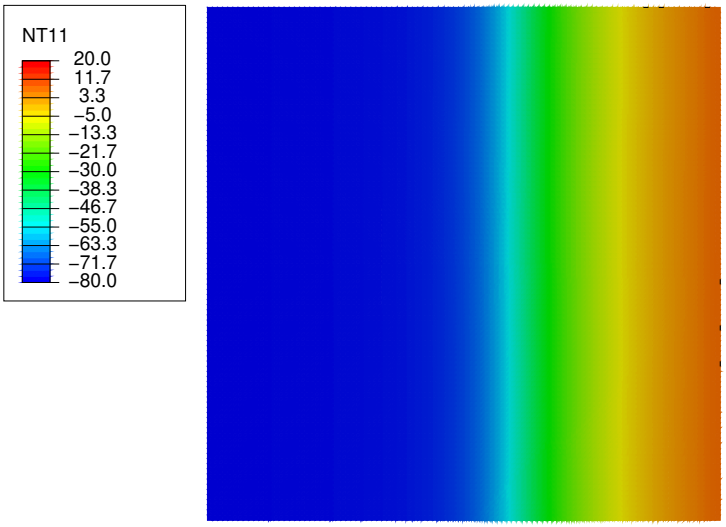


Figure 4: Electric potential at t = 500 ms.

Mechanical passive response

Elements tested

C3D8R

Features tested

Mechanical passive response of the cardiac tissue model.

Problem description

The test verifies the passive mechanical response of the cardiac tissue model. A unit cube is subjected to unaxial tension without fiber activation. Therefore, the total mechanical response is governed by the Holzapfel-Ogden model (see *Mechanical Passive Tissue Response* for additional details). While there are no analytical solutions for the uniaxial stress and lateral strains, they can be solved for numerically.

The model consists of a single C3D8R element of unit size. *Table 6* and *Table 7* list the material parameters used in the test. Given that there is no fiber activation (T_{max} = 0 MPa), the other parameters for the active response given in *Table 8* do not influence the solution.

Material: The material parameters used in the tests are given below.

Table 6: Fiber and sheet orientations.

Fiber direction	$(\cos(\theta), \sin(\theta), 0)$ with θ=49.98°.
Sheet direction	$(-\sin(\theta), \cos(\theta), 0)$ with θ=49.98°.

Table 7: Passive response material parameters.

a	0.004 MPa
b	8.0
a_f	0.005 MPa
b_f	5.0
a_s	0.002 MPa
b_s	2.0
a_{fs}	0.0 MPa
b_{fs}	0.0
D	0.01 MPa^{-1}
λ_{fL}	0

Table 8: Active response material parameters.

t_0	0.1 s
c_1	1048.9 s/mm
c_2	−1.429 s
l_0	0.00158 mm
B	4750 mm^{-1}
$Ca_{0_{max}}$	4.35 μmol/l

Ca_0	4.35 μmol/l
T_{max}	0.0 MPa
n	0.0
ϕ_{act}	15 mV
l_r	0.00185 mm
k	0

Results and discussion

The uniaxial stress, S_{11}, and lateral strains, LE_{22} and LE_{33}, obtained using the special-purpose material in Abaqus/Explicit agree well with the corresponding results obtained using an independent numerical method. All the results are shown in *Figure 5* and *Figure 6*.

Input files

cardiac_mechanical_passive.inp Mechanical passive response of the cardiac tissue model.

Figures

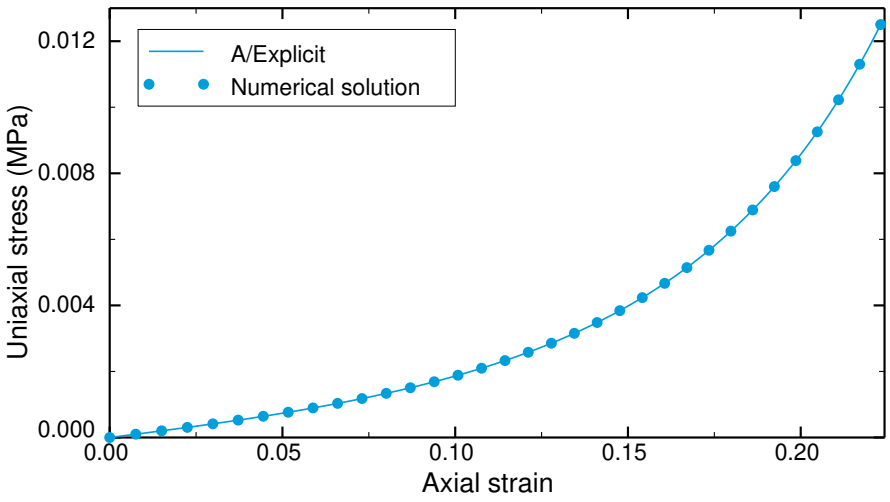


Figure 5: Uniaxial stress vs. axial strain.

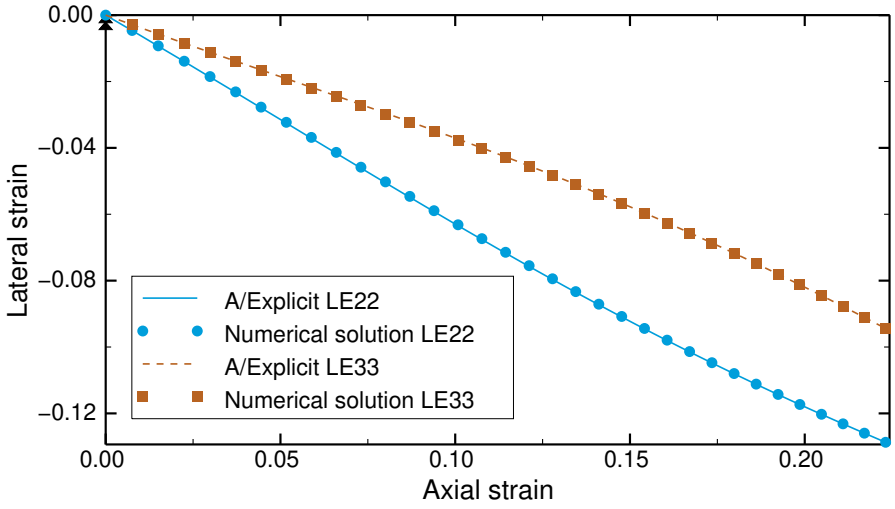


Figure 6: Lateral strains vs. axial strain.

Mechanical active response

Elements tested

C3D8R

Features tested

Mechanical active response of the cardiac tissue model.

Problem description

A unit cube is used to verify the active mechanical response of the cardiac tissue model. The cube is held fixed, and the electric potential is raised sufficiently for the material to generate an active stress. Given that the deformation is held constant, the time varying active stress can be calculated analytically from the active stress law used in the model (see *Mechanical Active Tissue Response* for additional details).

The model consists of a single C3D8R element of unit size. All displacement degrees of freedom are fixed. *Table 6*, *Table 7*, and *Table 9* list the material parameters used in the test. The activation potential is set to ϕ_{act} = 15 mV (the same as in the single-element test of the electrical model for cardiac tissue; see *Electrical conduction in the absence of diffusion*) and $\mathbf{k}$ = 1; therefore, field variable $\mathbf{FV_1}$ is used for activation. The model is simulated with an explicit dynamic step of T_{mech} = 0.7 s.

In a first test, the field variable $\mathbf{FV_1}$ is imported from the results of the electrical conduction simulation for the material with high refractoriness (see *Electrical conduction in the absence of diffusion*). For demonstration purposes, only results for times t_{elec} up to T_{elec} = 400 ms of that electrical simulation are used. With time scaling activated, the time t_{elec} of the electrical simulation is mapped to the time t_{mech} of the mechanical simulation according to

$$\frac{t_{mech}}{T_{mech}} = \frac{t_{elec}}{T_{elec}}.$$

Consequently, the activation time $t_{act,elec}$ = 13.5 ms from the electrical simulation (see *Table 3*) is expected to map to an activation time of

$$t_{act,mech} = \frac{T_{mech}}{T_{elec}} t_{act,elec} = 0.023625\,\mathrm{s}$$

Equation 1

in the mechanical simulation. For the chosen material properties, the total time needed for the active stress to develop and subside can be estimated to be no more than

$$t_{act,mech} + t_0 + t_r(0) = 0.63509\,\mathrm{s},$$

which is shorter than the total simulation time T_{mech} = 0.7 s.

In a second test, a simple piecewise-linear time history of the field variable $\mathbf{FV_1}$ is prescribed in such a way that the activation time matches the estimate $t_{act,mech}$ in *Equation 1*.

Another test verifies the successive activations of the active stress. The simulation consists of the second test repeated after itself. The simulation time is twice as long (T_{mech} = 1.4 s), and the field variable $\mathbf{FV_1}$ is prescribed so that the two successive activations occur at

$$t_{act}^{(1)} = t_{act,mech} = 0.023625\,\mathrm{s} \text{ and } t_{act}^{(2)} = t_{act}^{(1)} + 0.7 = 0.723625\,\mathrm{s}$$

Equation 2

Material: The material parameters used in the tests are given below.

Table 9: Active response material parameters.

t_0	0.1 s
c_1	1048.9 s/mm
c_2	-1.429 s
l_0	0.00158 mm
B	4750 mm^{-1}
$Ca_{0_{max}}$	4.35 μmol/l
Ca_0	4.35 μmol/l
$T_{\max}$	0.1 MPa
n	0.0
ϕ_{act}	15 mV
l_r	0.00185 mm
k	1

Results and discussion

The active stress obtained in each test using the special-purpose material in Abaqus/Explicit agrees with the analytical solution.

For the two tests with only one activation, the comparison is shown in *Figure 7*. The corresponding time histories of the activation field variable $\mathbf{FV_1}$ are shown in *Figure 8*, together with the time-scaled time history of the electric potential from the electrical simulation. The activation times in both simulations are reported in *Table*

10 and are consistent with *Figure 8*. The small discrepancies are attributed to slight differences in time integration schemes and temporal resolution.

For the test with the two successive activations, *Figure 9* shows the comparison between the active stress obtained in Abaqus/Explicit and the exact solution. *Figure 10* shows the corresponding activation indicator function. It shows that the first activation starts at $t = 0.0236252$ s and the second activation starts at $t = 0.723625$ s. This is consistent with the expected activation times in *Equation 2*.

Input files

cardiac_mechanical_active_imported.inp	Mechanical active response of the cardiac tissue model. The activation field variable is imported from the results of the electrical conduction simulation above for the material with high refractoriness.
cardiac_mechanical_active_prescribed.inp	Mechanical active response of the cardiac tissue model. The activation field variable is prescribed.

Tables

Table 10: Activation times for the single-activation tests.

Estimated	0.023625 s
Imported activation	0.023396 s
Prescribed activation	0.023625 s

Figures

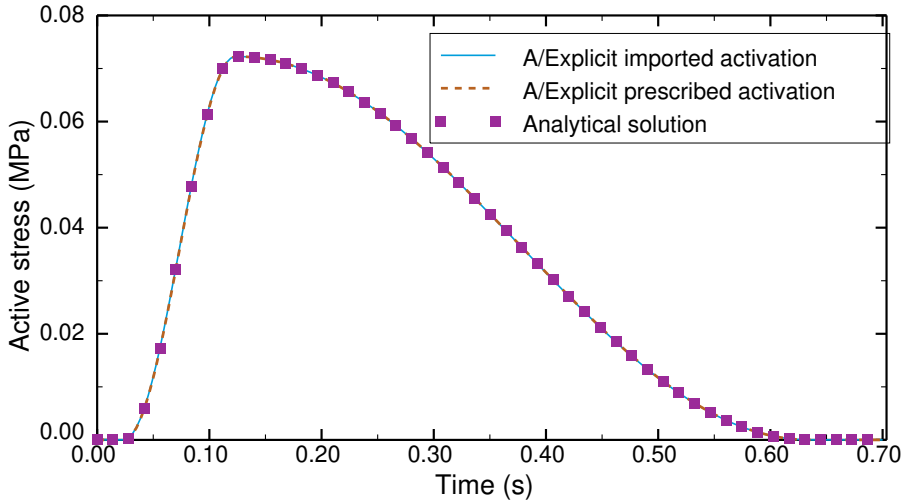


Figure 7: Active stress vs. time for the single-activation tests.

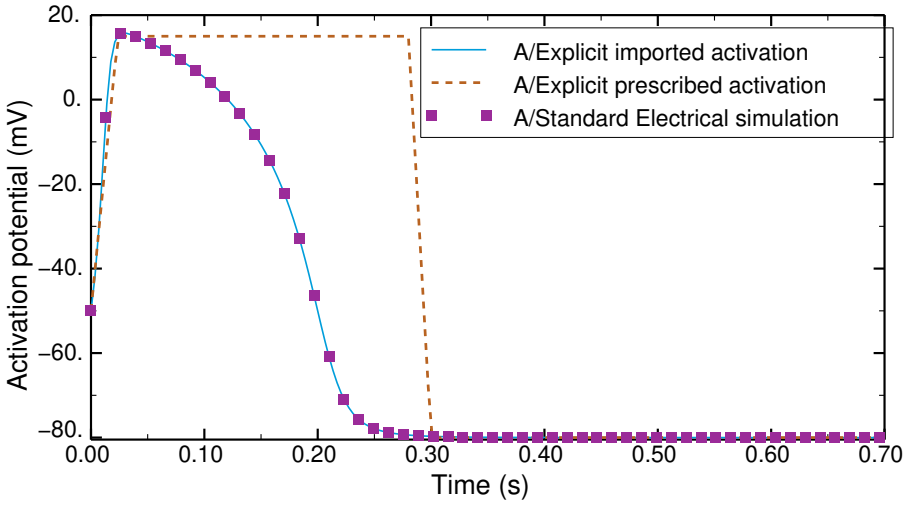


Figure 8: Activation potential vs. time for the single-activation tests.

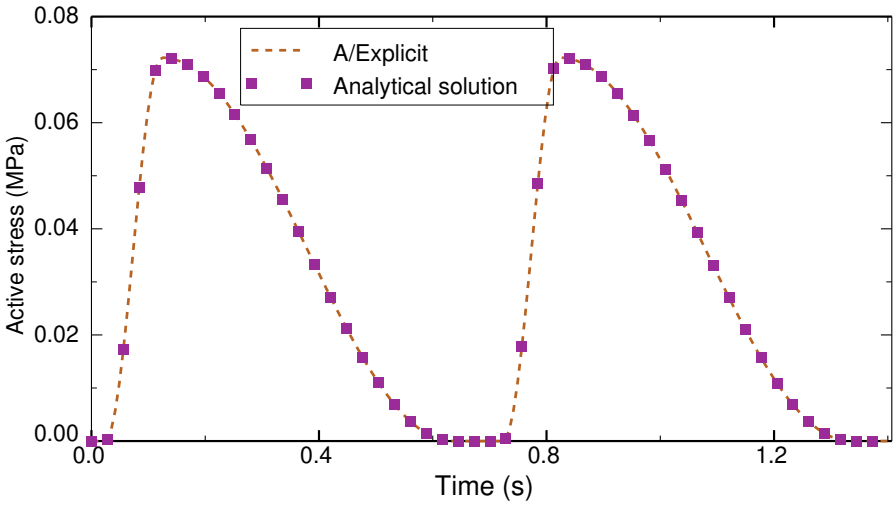


Figure 9: Active stress vs. time for the double-activation test.

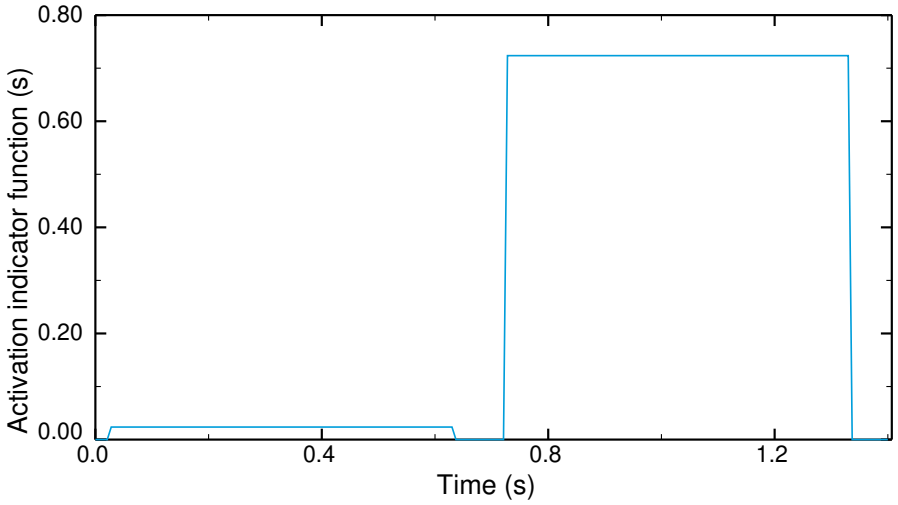


Figure 10: Activation indicator function vs. time for the double-activation test.

Thermal properties

In this section:

- *Field-variable-dependent conductivity*
- *Conductivity and specific heat*
- *Gap conductance*

Field-variable-dependent conductivity

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D8HT C3D8RHT C3D8RT C3D8T C3D10MHT C3D10MT C3D20HT C3D20RHT C3D20RT C3D20T CAX4HT CAX4RHT CAX4RT CAX4T CAX6MHT CAX6MT CGAX4HT CGAX4RHT CGAX4RT CGAX4T CGAX6MHT CGAX6MT CPE4HT CPE4RHT CPE4RT CPE4T CPE6MHT CPE6MT CPE8HT CPE8RHT CPE8RT CPE8T CPEG3T CPEG4HT CPEG4RHT CPEG4RT CPEG4T CPEG6MHT CPEG6MT CPEG8HT CPEG8RHT CPEG8T CPS4RT CPS4T CPS6MT DC3D8 DC3D10 DC3D20 DC2D3 DC2D4 DC2D6 DC2D8 DC1D2

Problem description

A one-dimensional steady-state heat transfer analysis with field-variable-dependent conductivity is performed. A heat rod with constant conductivity is placed on each side of a heat rod whose conductivity is a function of predefined field variables. These field variables are varied linearly over the course of the four increments of the analysis.

Model: Element 1: length = 1.0, area = 3.0, conductivity = 150.0

Element 2: length = 2.0, area = 3.0, conductivity = field-variable-dependent (see below)

Element 3: length = 3.0, area = 3.0, conductivity = 150.0

In Abaqus/Standard steady-state simulations are performed using both coupled temperature-displacement elements and pure heat transfer elements to model the rods. In Abaqus/ExplicitCPE4RT elements are used to model the heat rods (unit width is assumed for each heat rod), and a transient analysis is performed. The total simulation time is 1.40×10^6. This provides enough time for the transient solution to reach steady-state conditions in this problem.

Boundary conditions: T_1=1000.0, T_4=0.0

Results and discussion

The temperatures on each end of the rod (nodes 2 and 3) are reported below. These temperatures match the exact results.

Input files

Abaqus/Standard input files

fvdepcond_std_c3d8ht.inp	Field-variable-dependent conductivity; C3D8HT elements.
fvdepcond_std_c3d8rht.inp	Field-variable-dependent conductivity; C3D8RHT elements.
fvdepcond_std_c3d8rt.inp	Field-variable-dependent conductivity; C3D8RT elements.

fvdepcond_std_c3d8t.inp	Field-variable-dependent conductivity; C3D8T elements.
fvdepcond_std_c3d10mht.inp	Field-variable-dependent conductivity; C3D10MHT elements.
fvdepcond_std_c3d10mt.inp	Field-variable-dependent conductivity; C3D10MT elements.
fvdepcond_std_c3d20ht.inp	Field-variable-dependent conductivity; C3D20HT elements.
fvdepcond_std_c3d20rht.inp	Field-variable-dependent conductivity; C3D20RHT elements.
fvdepcond_std_c3d20rt.inp	Field-variable-dependent conductivity; C3D20RT elements.
fvdepcond_std_c3d20rt_post.inp	Field-variable-dependent conductivity; **POST OUTPUT* analysis.
fvdepcond_std_c3d20t.inp	Field-variable-dependent conductivity; C3D20T elements.
fvdepcond_std_cax4ht.inp	Field-variable-dependent conductivity; CAX4HT elements.
fvdepcond_std_cax4rht.inp	Field-variable-dependent conductivity; CAX4RHT elements.
fvdepcond_std_cax4rt.inp	Field-variable-dependent conductivity; CAX4RT elements.
fvdepcond_std_cax4t.inp	Field-variable-dependent conductivity; CAX4T elements.
fvdepcond_std_cax6mht.inp	Field-variable-dependent conductivity; CAX6MHT elements.
fvdepcond_std_cax6mt.inp	Field-variable-dependent conductivity; CAX6MT elements.
fvdepcond_std_cgax4ht.inp	Field-variable-dependent conductivity; CGAX4HT elements.
fvdepcond_std_cgax4rht.inp	Field-variable-dependent conductivity; CGAX4RHT elements.
fvdepcond_std_cgax4rt.inp	Field-variable-dependent conductivity; CGAX4RT elements.
fvdepcond_std_cgax4t.inp	Field-variable-dependent conductivity; CGAX4T elements.
fvdepcond_std_cgax6mht.inp	Field-variable-dependent conductivity; CGAX6MHT elements.
fvdepcond_std_cgax6mt.inp	Field-variable-dependent conductivity; CGAX6MT elements.
fvdepcond_std_cpe4ht.inp	Field-variable-dependent conductivity; CPE4HT elements.
fvdepcond_std_cpe4rht.inp	Field-variable-dependent conductivity; CPE4RHT elements.
fvdepcond_std_cpe4rt.inp	Field-variable-dependent conductivity; CPE4RT elements.
fvdepcond_std_cpe4t.inp	Field-variable-dependent conductivity; CPE4T elements.
fvdepcond_std_cpe6mht.inp	Field-variable-dependent conductivity; CPE6MHT elements.
fvdepcond_std_cpe6mt.inp	Field-variable-dependent conductivity; CPE6MT elements.
fvdepcond_std_cpe8ht.inp	Field-variable-dependent conductivity; CPE8HT elements.
fvdepcond_std_cpe8rht.inp	Field-variable-dependent conductivity; CPE8RHT elements.
fvdepcond_std_cpe8rt.inp	Field-variable-dependent conductivity; CPE8RT elements.
fvdepcond_std_cpe8t.inp	Field-variable-dependent conductivity; CPE8T elements.
fvdepcond_std_cpeg3t.inp	Field-variable-dependent conductivity; CPEG3T elements.
fvdepcond_std_cpeg4ht.inp	Field-variable-dependent conductivity; CPEG4HT elements.
fvdepcond_std_cpeg4rht.inp	Field-variable-dependent conductivity; CPEG4RHT elements.
fvdepcond_std_cpeg4rt.inp	Field-variable-dependent conductivity; CPEG4RT elements.
fvdepcond_std_cpeg4t.inp	Field-variable-dependent conductivity; CPEG4T elements.
fvdepcond_std_cpeg6mht.inp	Field-variable-dependent conductivity; CPEG6MHT elements.
fvdepcond_std_cpeg6mt.inp	Field-variable-dependent conductivity; CPEG6MT elements.
fvdepcond_std_cpeg8ht.inp	Field-variable-dependent conductivity; CPEG8HT elements.
fvdepcond_std_cpeg8rht.inp	Field-variable-dependent conductivity; CPEG8RHT elements.
fvdepcond_std_cpeg8t.inp	Field-variable-dependent conductivity; CPEG8T elements.
fvdepcond_std_cps4rt.inp	Field-variable-dependent conductivity; CPS4RT elements.
fvdepcond_std_cps4t.inp	Field-variable-dependent conductivity; CPS4T elements.
fvdepcond_std_cps6mt.inp	Field-variable-dependent conductivity; CPS6MT elements.

mcdisd1nt1.inp	Field-variable-dependent conductivity; DC1D2 elements.
fvdepcond_std_dc3d8.inp	Field-variable-dependent conductivity; DC3D8 elements.
fvdepcond_std_dc3d10.inp	Field-variable-dependent conductivity; DC3D10 elements.
fvdepcond_std_dc3d20.inp	Field-variable-dependent conductivity; DC3D20 elements.
fvdepcond_std_dc2d3.inp	Field-variable-dependent conductivity; DC2D3 elements.
fvdepcond_std_dc2d4.inp	Field-variable-dependent conductivity; DC2D4 elements.
fvdepcond_std_dc2d6.inp	Field-variable-dependent conductivity; DC2D6 elements.
fvdepcond_std_dc2d8.inp	Field-variable-dependent conductivity; DC2D8 elements.

Abaqus/Explicit input file

fvdepcond_xpl_cpe4rt.inp	Field-variable-dependent conductivity; CPE4RT elements.

Conductivity and specific heat

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

CPE4T CPE4RT CPEG4T DC1D3

Problem description

A simple transient heat transfer analysis of a heat link constructed with DC1D3 elements is considered in Abaqus/Standard. In Abaqus/ExplicitCPE4RT elements are used to model the heat link. The temperature at one end of the link is fixed, while a flux is applied to the other end. The conductivity and the specific heat of the material comprising the heat link vary with prescribed values of a field variable (FV). The value of this field variable is altered with time.

In both Abaqus/Standard and Abaqus/Explicit a transient analysis is conducted. The total simulation time is 6.

Results and discussion

The nodal temperatures of the link confirm that the thermal properties of the material do, indeed, depend on the field variable. The actual values of the material parameters as a function of the field variable are, therefore, correct, since the temperatures were calculated from these parameters by Abaqus.

Input files

Abaqus/Standard input files

mcsisd1nt1.inp	Conductivity and specific heat analysis; DC1D3 elements.
fvcondspec_std_cpe4t.inp	Conductivity and specific heat analysis; CPE4T elements.
fvcondspec_std_cpeg4t.inp	Conductivity and specific heat analysis; CPEG4T elements.

Abaqus/Explicit input file

fvcondspec_xpl_cpe4rt.inp	Conductivity and specific heat analysis; CPE4RT elements.

Gap conductance

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

CPEG4T C3D8RT C3D8T DC3D8 DC3D10 DCC3D8 SC8RT S4RT

Problem description

The tests in this section are set up as cases of uniform one-dimensional heat flux using generalized plane strain (Abaqus/Standard only), and three-dimensional elements. In all Abaqus/Standard cases a steady-state heat transfer analysis is performed. In Abaqus/Explicit a transient analysis is performed for each case, with a simulation time chosen to ensure that steady-state conditions are reached in this problem. Particular values (gap clearance, predefined field variables, etc.) vary during the solution, which in turn influence the conductivity across the interface and, thus, the solution.

Results and discussion

The results match the exact solutions.

Input files

Abaqus/Standard input files

mgcgco1ctug.inp	Gap clearance-dependent conductivity, CPEG4T elements.
mgcgpo1ctug.inp	Gap pressure-dependent conductivity, CPEG4T elements.
mgcgco1ctus.inp	Gap clearance-dependent conductivity, C3D8T elements.
mgcgpo1ctus.inp	Gap pressure-dependent conductivity, C3D8T elements.
mgcgcd1ctus.inp	Field-variable-dependent conductivity, DC3D8 elements.
mgcoot1hts.inp	Gap temperature-dependent conductivity, DC3D8 elements.
mgcood1hts.inp	Gap field-variable-dependent conductivity, DC3D8 elements.
mgcmfo1hts.inp	Gap mass-flow-rate-dependent conductivity, DCC3D8 elements.
mdc3d10_tdgc_stos_hts.inp	Gap temperature-dependent conductivity, DC3D10 elements.

Abaqus/Explicit input files

gapclearcond_x_c3d8rt.inp	Gap clearance-dependent conductivity, C3D8RT elements.
gapclearcond_x_sc8rt.inp	Gap clearance-dependent conductivity, SC8RT elements.
gappresscond_x_c3d8rt.inp	Gap pressure-dependent conductivity, C3D8RT elements.
gappresscond_x_sc8rt.inp	Gap pressure-dependent conductivity, SC8RT elements.
gapfvcond_x_c3d8rt.inp	Field-variable-dependent conductivity using contact pairs, C3D8RT elements.

gapfvcond_x_gcont_c3d8rt.inp	Field-variable-dependent conductivity using general contact, C3D8RT elements.
gapclearcond_x_c3d8t.inp	Gap clearance-dependent conductivity, C3D8T elements.
gappresscond_x_c3d8t.inp	Gap pressure-dependent conductivity using contact pairs, C3D8T elements.
gappresscond_x_gcont_c3d8t.inp	Gap pressure-dependent conductivity using general contact, C3D8T elements.
gapfvcond_x_c3d8t.inp	Field-variable-dependent conductivity, C3D8T elements.
gapclearanl_x_gcont_c3d8rt.inp	Gap clearance-dependent conductivity with analytical rigid surface using general contact, C3D8RT elements.
gapclearcond_x_gcont_s4rt.inp	Gap clearance-dependent conductivity using general contact, S4RT elements.

Piezoresistivity properties: Piezoresistive effect

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

Q3D8 QAX4

Problem description

The tests in this section model one-dimensional electrical current flow using axisymmetric and three-dimensional elements. In all cases a steady-state thermal-electrical-structural analysis is performed.

Results and discussion

The results match the exact solutions.

Input files

Input files

prer38vfdf.inp	Piezoresistivity, Q3D8 elements.
prera4vfdf.inp	Piezoresistivity, QAX4 elements.

Output Verification

In this section:

- *Writing element matrix output records*
- *Substructure matrix output request*
- *Integrated output variables*
- *Rigid body motion output variables*
- *Element nodal forces in beam section orientation*

Writing element matrix output records

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

B21 C3D8 DS3 DS4
S4 S4R S4R5 S8R

Features tested

The output and input of element matrices are tested.

Problem description

These tests verify that the matrices written out by the element matrix output procedure are valid and that they can be input into an analysis and used again. The validity of the element matrices is tested by an analysis that uses the matrices to solve a linear problem.

Results and discussion

xemob21o.inp, xemob21u.inp

The maximum displacement in this problem is 332.04. The computed displacements in both problems match this value, and the displacements at the other nodes match as well.

xemoc38o.inp, xemoc38u.inp

The maximum displacement of −2.0E−4 occurs at nodes 3 and 7 in this problem. Both runs have identical displacement fields.

xemods3o.inp, xemods4u.inp

The temperature variation through the plate in this example is the same at all nodes. The bottom has a temperature of 0.0, the middle temperature is 746.0, and the top has a temperature of 994.7. The results for both cases are the same.

xemods4o.inp, xemods4u.inp

The temperature variation through the plate in this example is the same at all nodes. The bottom has a temperature of 0.0, the middle temperature is 746.2, and the top has a temperature of 994.69. The results for both runs are the same.

xemos45o.inp, xemos45u.inp

The maximum displacement of −8.6667E−5 occurs at node 4 in both problems. However, the problem which uses the previously computed matrices is missing the rotation at node 1. This extra degree of freedom in the first run is a result of special procedures that activate the rotation if a boundary or loading condition is applied there. The precomputed element stiffness matrix does not have this capability.

xemos4o.inp, xemos4u.inp

The maximum displacement of −8.6667E−5 occurs at node 4 in both problems. Both runs have identical displacement fields, including the rotations. There is no conditional activation of rotation degrees of freedom with the S4 elements as there is with the S4R5 elements.

xemos4ro.inp, xemos4ru.inp

The maximum displacement of −8.6667E−5 occurs at node 4 in both problems. Both runs have identical displacement fields, including the rotations. There is no conditional activation of rotation degrees of freedom with the S4R elements, as there is with the S4R5 elements.

xemos8ro.inp, xemos8ru.inp

The maximum displacement of −3.6376 occurs at node 89. Both runs have the same displacements.

Input files

Output of element matrix:

xemob21o.inp	Cantilever made up of five B21 elements with a point load at the end. A static step is used to test the matrices.
xemoc38o.inp	One C3D8 element with distributed and concentrated loads. A static step is used to test the matrix.
xemods3o.inp	One DS3 element with a distributed flux. A steady-state heat transfer step is used to test the matrix.
xemods4o.inp	One DS4 element with a distributed flux. A steady-state heat transfer step is used to test the matrix.
xemos45o.inp	One S4R5 element with concentrated loads. A static step is used to test the matrix.
xemos4o.inp	One S4 element with concentrated loads. A static step is used to test the matrix.
xemos4ro.inp	One S4R element with concentrated loads. A static step is used to test the matrix.
xemos8ro.inp	One S8R element with distributed loads. A static step is used to test the matrix.

Input of element matrix:

xemob21u.inp	Cantilever made up of five B21 elements with a point load at the end. A static step is used to test the matrices.
xemoc38u.inp	One C3D8 element with distributed and concentrated loads. A static step is used to test the matrix.
xemods3u.inp	One DS3 element with a distributed flux. A steady-state heat transfer step is used to test the matrix.
xemods4u.inp	One DS4 element with a distributed flux. A steady-state heat transfer step is used to test the matrix.

xemos45u.inp	One S4R5 element with concentrated loads. A static step is used to test the matrix.
xemos4u.inp	One S4 element with concentrated loads. A static step is used to test the matrix.
xemos4ru.inp	One S4R element with concentrated loads. A static step is used to test the matrix.
xemos8ru.inp	One S8R element with distributed loads. A static step is used to test the matrix.

Substructure matrix output request

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D8 Substructures and linear user elements.

Features tested

The output and input of substructure matrices and load case vectors are tested.

Problem description

These tests verify that the matrices and load vectors written out by the substructure matrix output option are valid and that they can be input into an analysis and used again. The validity of the results is tested by an analysis that uses the generated matrices and load vectors to solve a linear problem.

Results and discussion

xsmon2so.inp, xsmon2su.inp

Input files *xsmon2so.inp* and *xsmon2su.inp* use the substructure and the associated matrices generated by the substructure generation analysis defined in input file *xsmon2so_gen1.inp*. In the model defined in *xsmon2so.inp*, a single substructure element is used and the generated substructure loads are activated. In the model defined in *xsmon2su.inp*, a single linear user element is defined using the stiffness matrix output from the generation job *xsmon2so_gen1.inp* and the model is loaded with a system of concentrated loads. Both runs yield identical displacement fields. The maximum displacement of -2.0×10^{-4} occurs at node 3 in both the runs.

xsmop1so.inp, xsmop1su.inp

Input files *xsmop1so.inp* and *xsmop1su.inp* use the substructure and the associated matrices generated by the substructure generation analysis defined in input file *xsmop1so_gen1.inp*. In the model defined in *xsmop1so.inp*, a single substructure element is used and the generated substructure loads are activated. In the model defined in *xsmop1su.inp*, a single linear user element is defined using the stiffness matrix output from the generation job *xsmop1so_gen1.inp* and the model is loaded with a system of concentrated loads. Both runs yield identical displacement fields. The maximum displacement of -2.0×10^{-4} occurs at node 3 in both the runs.

Input files

Output of substructure matrices and loads:

xsmon2so_gen1.inp A substructure made up of one C3D8 element with distributed loads in one **SUBSTRUCTURE LOAD CASE* and concentrated loads in another.

xsmop1so_gen1.inp A substructure made up of one C3D8 element with concentrated loads applied in a preloading step (for demonstration purpose) and distributed loads applied in a **SUBSTRUCTURE LOAD CASE*.

Using substructures and substructure loads:

xsmon2so.inp This file uses the substructure matrix and load vectors generated using *xsmon2so_gen1.inp*. A static step is used to test the matrix and load vectors.

xsmop1so.inp This file uses the substructure matrix and load vectors generated using *xsmop1so_gen1.inp*. A static step is used to test the matrix and load vectors.

Using equivalent user elements:

xsmon2su.inp The user element is created using the substructure matrix generated using *xsmon2so_gen1.inp*. A static step is used to test the matrix when the load vectors are applied.

xsmop1su.inp The user element is created using the substructure matrix generated using *xsmop1so_gen1.inp*. A static step is used to test the matrix when the load vectors are applied.

Integrated output variables

Products: Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

B21 B22 B31 B32
PIPE21 PIPE31
C3D4 C3D6 C3D8 C3D8I C3D8R C3D10M
CPE4R CPE6M CPS3 CPS4R
M3D4R
S4 S4R SC8R
T2D2 T3D2

Features tested

Output variables SOF and SOM give the total force and total moment transmitted across a given surface. This surface typically forms a cross-section cutting through a deformable continuum or structure. The area of the specified surface when projected along the average normal to that surface is given by output variable SOAREA. The vector output is given in the global basis, and the total moment is taken about the global origin by default. However, an integrated output section can be defined and associated with the integrated output request to obtain the output in a moving coordinate system and the total moment taken about an anchor point that may be translating and/or rotating.

Problem description

The integrated output variables can be requested only as history output to the output database. These variables are considered *whole element set variables*, meaning that the quantity requested is summed over the facets of the elements lying under the surface specified.

Each of the verification problems below models a region of given element type, and a number of cross-section-like surfaces are defined. A uniform initial stress is specified for the entire region. All the nodes of the region are then included in a rigid body that is constrained to undergo a large rotation. Under this rigid body motion the stresses should remain constant. Hence, the total force and the total moment vectors should correspond to the initial stresses and also remain constant. In addition, the integrated output is tested over surfaces through integrated output section definitions.

Results and discussion

These verification problems all impose a simple rigid body motion, and each contains the material under a specified initial stress. In all cases the integrated output based on the fixed stresses remains constant.

Input files

integratedoutput_b21.inp	Integrated output over a surface formed on the ends of a beam.
integratedoutput_b22.inp	Integrated output over a surface formed on the ends of a beam.

integratedoutput_b31.inp	Integrated output over a surface formed on the ends of a beam.
integratedoutput_b32.inp	Integrated output over a surface formed on the ends of a beam.
integratedoutput_p21.inp	Integrated output over a surface formed on the ends of a pipe.
integratedoutput_p31.inp	Integrated output over a surface formed on the ends of a pipe.
integratedoutput_c3d4.inp	Integrated output over a surface cutting through a solid.
integratedoutput_c3d6.inp	Integrated output over a surface cutting through a solid.
integratedoutput_c3d8.inp	Integrated output over a surface cutting through a solid.
integratedoutput_c3d8i.inp	Integrated output over a surface cutting through a solid.
integratedoutput_c3d8r.inp	Integrated output over a surface cutting through a solid.
integratedoutput_cpe4r.inp	Integrated output over a surface cutting through a solid.
integratedoutput_cpe6m.inp	Integrated output over a surface cutting through a solid.
integratedoutput_cps3.inp	Integrated output over a surface cutting through a solid.
integratedoutput_cps4r.inp	Integrated output over a surface cutting through a solid.
integratedoutput_m3d4r.inp	Integrated output over a surface cutting through a membrane.
integratedoutput_s4.inp	Integrated output over a surface cutting through a shell.
integratedoutput_s4r.inp	Integrated output over a surface cutting through a shell.
integratedoutput_sc8r.inp	Integrated output over a surface cutting through a continuum shell.
integratedoutput_t2d2.inp	Integrated output over a surface formed on the ends of a truss.
integratedoutput_t3d2.inp	Integrated output over a surface formed on the ends of a truss.

Rigid body motion output variables

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

B21 B21H B22 B22H B23 B23H B31 B31H B32 B32H B33 B33H
C3D4 C3D6 C3D8 C3D10 C3D15 C3D20
CAX3 CAX4 CAX4R CAX6 CAX8 CAX8R
CPEG3 CPEG3H CPEG4 CPEG4H CPEG4R CPEG4RH CPEG4I CPEG4IH CPEG6 CPEG6H CPEG8 CPEG8H CPEG8R CPEG8RH
CPE3 CPE3H CPE4 CPE4H CPE4R CPE4RH CPE4I CPE4IH CPE6 CPE6H
CPE8 CPE8H CPE8R CPE8RH
CPS3 CPS4 CPS4R CPS6 CPS8 CPS8R
ELBOW31 ELBOW31B ELBOW31C ELBOW32
M3D3 M3D4 M3D4R M3D6 M3D8 M3D8R M3D9 M3D9R MASS
PIPE21 PIPE21H PIPE31 PIPE31H PIPE32 PIPE32H ROTARYI PIPE22 PIPE22H
S3R S4 S4R S4R5 S8R S8R5 S9R5 STRI3 STRI65
SAX1 SAX2 SPRING1

Features tested

The following output variables describe the equivalent rigid body motion for any general dynamic motion:

- current coordinate (XC),
- displacement of the center of mass UC (URC),
- equivalent rigid body velocity components VC (VRC),
- current angular momentum about the center of mass (HC),
- current angular momentum about the center of origin (HO),
- current rotary inertia (RI),
- current mass (MASS), and
- current volume (VOL).

These output variables are valid only for direct-integration implicit dynamic procedures. The accuracy of these output variables is verified with a test suite that encompasses all elements that have mass and/or rotary inertia.

Problem description

The equivalent rigid body motion output variables are specified in the results (`.fil`) and data (`.dat`) files. They can only be requested when using the direct-integration implicit dynamic procedure. These variables are considered *whole element set variables*, meaning that the quantity requested is summed over the element set specified. If no element set is specified, the quantity is summed over the entire model. The element set specified may contain elements which do not have mass (SPRINGs, DASHPOTs, etc.), but these elements will be ignored

during the summation process. Specifying an element set in which all elements have no mass will elicit a warning message from Abaqus.

All of the verification problems below impose a rigid body motion on single element models. Each input file contains separate and distinct single element meshes corresponding to the many specific elements within that element category. For instance, the *xrbmcpes.inp* input file tests all of the CPE type elements and contains single element meshes for the CPE3, CPE4, CPE4R, CPE6, CPE8, CPE8R elements (and hybrid versions of all these elements). Most of the problems impose a planar 90° rotation about the z-axis; the three-dimensional continuum problem imposes an oblique rotation. Separate output requests to the results file are given for each element set in the model.

Results and discussion

These verification problems all impose a simple rigid body motion. In all cases the magnitude of the rigid body output variables should agree with the imposed motion. For some problems (such as those with an imposed constant velocity) the expected magnitudes of the output variables can be calculated directly from the imposed motion. In other problems the expected output variable magnitudes can be calculated from the imposed motion and the element geometry.

Input files

xrbmaxis.inp	Tests all axisymmetric elements, including axisymmetric shells. A constant z-velocity is imposed (there is no valid rotation in axisymmetric problems).
xrbmbeam.inp	Tests all beam elements (excludes open section beams). A rigid 90° rotation is imposed about the z-axis.
xrbmbepo.inp	Uses results of xrbmbeam.inp to verify the **POST OUTPUT* option.
xrbmt3ds.inp	Tests all three-dimensional truss elements.
xrbmc3ds.inp	Tests all three-dimensional continuum elements. A rigid rotation is imposed about the direction (.707, .707, 0.0).
xrbmcpeg.inp	Tests all generalized plane strain elements.
xrbmcpes.inp	Tests all plane strain elements.
xrbmcper.inp	Restart of xrbmcpes.inp. Tests **RESTART* without END STEP.
xrbmcpss.inp	Tests all plane stress elements.
xrbmelbw.inp	Tests all elbow elements.
xrbmmass.inp	Tests the mass element. A constant x-velocity is imposed. Tests are done with and without the **TRANSFORM* option.
xrbmmemb.inp	Tests all membrane elements. A rigid rotation is imposed about the z-axis. Shell elements overlay the membranes for stability purposes.
xrbmroti.inp	Four separate tests of the rotary inertia element. The first test uses a constant velocity. The second imposes both a translation and rotation. The third is similar to the second, but adds the use of the **TRANSFORM* option. The fourth test is similar to the second, but adds both **TRANSFORM* and **ORIENTATION*.
xrbmshel.inp	Tests all shell elements. A rigid 90° rotation is imposed about the z-axis.

The following tests contain two distinct element types:

xrbmaxb1.inp	This model consists of an axisymmetric solid and a three-dimensional beam element. A constant y-velocity is imposed. A rigid body output request is made for the axisymmetric element, and another is made for the whole model. This tests the format of the printed output. The axisymmetric printed output is limited to specific directions. With the addition of a three-dimensional beam, the whole model output must be given for all directions.

xrbmaxb2.inp	This test is similar to xrbmaxb1.inp, but the order of the element generation is reversed. The printed output should be identical to the output for xrbmaxb1.inp.
xrbmaxb3.inp	This test is similar to xrbmaxb1.inp but uses a two-dimensional beam element. Again, the purpose is to compare the format of the printed output for the axisymmetric element to the output for the whole model.
xrbmsprg.inp	This model consists of a spring and a beam. A rigid body variable request is made for an element set containing only the spring. This should trigger a warning message from Abaqus. Another output request is made for the whole model. The output should agree with the imposed rigid rotation of the beam element.

Element nodal forces in beam section orientation

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

B21 B22 B23 B31 B32 B33

Features tested

Output variable NFORCSO gives the element nodal forces caused by stress in the element in the same coordinate system used to output section forces and moments. NFORCSO differs from NFORC only in the coordinate system used for output: NFORCSO components are the internal forces in the beam coordinate system, while NFORC components are internal forces in the global coordinate system.

Problem description

An L-shaped cantilever beam has concentrated loads applied at its free end. The length of each segment is 10 in., and the beam has a square cross-section with 0.10 in. sides. Steel elastic material properties are used (Young's modulus of 30×10^6 psi and Poisson's ratio of 0.3). Since the beam is slender, we choose to have the slenderness correction coefficient (SCF) computed from the elastic material definition; by adding the label SCF in the transverse shear stiffness definition, we obtain improved results with the linear Timoshenko beam elements B21 and B31.

Results and discussion

The results illustrate how the variable NFORCSO provides a more convenient way of examining results along beams, especially the case in long linear Timoshenko beam elements, since these elements possess a single integration point along the length of the beam. Output variable NFORCSO provides the bending moments at the extremities of the element, thus depicting the linearly varying bending moment distributions in the problem at hand. In quadratic Timoshenko beam elements B22 and B32 all NFORCSO components vanish at the center nodes as expected.

Input files

nforcso_b21.inp	B21 elements.
nforcso_b22.inp	B22 elements.
nforcso_b23.inp	B23 elements.
nforcso_b31.inp	B31 elements.
nforcso_b32.inp	B32 elements.
nforcso_b33.inp	B33 elements.

User Subroutines Verification

In this section:

- *About user subroutine verification tests*
- *User subroutine verification*

About user subroutine verification tests

Overview

This section contains test cases that provide evidence that the implementation of user subroutines produces results that can be compared to independently calculated solutions or experimental results.

User subroutines are provided to increase the functionality of several Abaqus capabilities for which the usual data input methods alone may be too restrictive.

Information on writing user subroutines and detailed descriptions of each subroutine appear online in *Abaqus User Subroutines Guide*. A listing and explanations of associated utility routines also appear in that guide.

References:

- *About User Subroutines and Utilities*

User subroutine verification

In this section:

- *DFLUX*
- *DISP*
- *DLOAD*
- *FRIC*
- *FRIC_COEF*
- *GAPCON*
- *GAPELECTR*
- *HARDINI*
- *HETVAL*
- *RSURFU*
- *SDVINI*
- *UAMP*
- *UANISOHYPER_INV and VUANISOHYPER_INV*
- *UCOHESIVEOFFSET*
- *UCREEPNETWORK*
- *UCURE*
- *UEL*
- *UELMAT*
- *UEPACTIVATIONVOL*
- *UEXPAN*
- *UFLUID*
- *UGENS*
- *UHARD*
- *UHYPER_STRETCH*
- *UINTER*
- *UMAT and UHYPER*
- *UMATHT*
- *URDFIL*
- *USDFLD*
- *UTEMP, VUTEMP, UFIELD, UMASFL, and UPRESS*
- *UTRSNETWORK*
- *UVAREL*
- *UVARM*
- *UWAVE and UEXTERNALDB*
- *VDISP*
- *VDLOAD*
- *VFILM*
- *VFRIC, VFRIC_COEF, and VFRICTION*
- *VHETVAL*
- *VSDVINI*

- *VUAMP*
- *VUCHARLENGTH*
- *VUCREEPNETWORK*
- *VUEL*
- *VUFIELD*
- *VUGENS*
- *VUHARD*
- *VUHYPER*
- *VUHYPER_STRETCH*
- *VUINTER*
- *VUINTERACTION*
- *VUMAT*
- *VUMATHT*
- *VUSDFLD*
- *VUTRSNETWORK*
- *VUVISCOSITY*
- *VWAVE*

DFLUX

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Features tested

User subroutine to define nonuniform distributed flux in heat transfer and mass diffusion analyses.

Heat transfer analysis

Elements tested

DC2D8

Problem description

A steady-state heat transfer analysis of a unit block is performed. The block is composed of six DC2D8 elements. Side *A* of the block (nodes 1–7) has its temperature, θ_A, ramped up linearly over the course of a step. The opposite side of the block, side *B* (nodes 201–207), has a nonuniform distributed flux, q_B, applied to it via user subroutine *DFLUX*. The value of the distributed flux varies as a function of the current temperature of this side, θ_B. This variation of applied flux is chosen to be $q_B(\theta_B) = -k\theta_B$, where *k* is the conductivity of the block material. A thermal energy balance,

$$q = -k\frac{\partial\theta}{\partial y} = -k\frac{(\theta_A-\theta_B)}{\Delta y} = q_B(\theta_B) = -k\theta_B,$$

gives us a solution for θ_B such that $\theta_B = \theta_A/2$.

The inclusion of $dq/d\theta_B$ in user subroutine *DFLUX* is essential for good convergence of the solution.

Results and discussion

The results match the exact solution.

Input files

udfluxxx.inp	Test of *DFLUX* in a heat transfer analysis.
udfluxxx.f	User subroutine *DFLUX* used in udfluxxx.inp.

Mass diffusion analysis

Elements tested

DC2D8

Problem description

A steady-state mass diffusion analysis of a unit block is performed. The block is composed of six DC2D8 elements. Side *A* of the block (nodes 1–7) has its normalized concentration, ϕ_A, ramped up linearly over the course of a step. The opposite side of the block, side *B* (nodes 201–207), has a nonuniform distributed flux, q_B, applied to it via user subroutine *DFLUX*. The value of the distributed flux varies as a function of the current normalized concentration, ϕ_B; temperature, θ_B; and equivalent pressure stress, p_B, of this side. This variation of applied flux is chosen to be $q_B(\phi_B, \theta_B, p_B) = -D(\theta, p)\phi_B$, where $D(\theta, p)$ is the diffusivity of the block material. The diffusivity is defined as

$$D(\theta, p) = D_0 + D_1\theta + D_2 p + D_3\theta p,$$

and diffusion is otherwise considered to be independent of temperature and equivalent pressure stress (i.e., $\kappa_p = \kappa_s = 0$). The temperature and pressure stress fields are specified at all nodes and are ramped up linearly over the course of the step. The mass balance,

$$q = -D(\theta, p)\frac{\partial \phi}{\partial y} = -D(\theta, p)\frac{(\phi_A - \phi_B)}{\Delta y} = q_B(\phi_B, \theta_B, p_b) = -D(\theta, p)\phi_B,$$

gives a solution for ϕ_B such that $\phi_B = \phi_A/2$.

The inclusion of $dq/d\phi_B$ in user subroutine *DFLUX* is essential for good convergence of the solution.

Results and discussion

The results match the exact solution.

Input files

udfluxmd.inp	Test of *DFLUX* in a mass diffusion analysis.
udfluxmd.f	User subroutine *DFLUX* used in udfluxmd.inp.

DISP

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

T3D2

Features tested

User subroutine to provide prescribed nodal behavior (displacements, velocities, and accelerations).

Problem description

A straight section built with one-dimensional truss elements is used in a dynamic analysis. The model has boundary conditions prescribed at nodes 2, 3, and 4 (nodes TRUSS.3, TRUSS.5, and TRUSS.7 in the model defined in terms of an assembly of part instances) using user subroutine *DISP*, while at nodes 5, 6, and 7 (TRUSS.9, TRUSS.11, and TRUSS.13) the boundary conditions are prescribed using an amplitude definition. A displacement variation is specified at nodes 2 and 5 (TRUSS.3 and TRUSS.9), a velocity variation is specified at nodes 3 and 6 (TRUSS.5 and TRUSS.11), and an acceleration variation is specified at nodes 4 and 7 (TRUSS.7 and TRUSS.13). The variation prescribed is

$$f(x) = 1.5 \quad \sin(.1\pi t) + .5\cos(.1\pi t).$$

For the variations specified using *DISP*, the appropriate derivatives and integrals have to be incorporated into the subroutine. For the amplitude specification Abaqus automatically performs the necessary differentiation and integration. Identical variations are specified in both methods such that the results should be identical.

Results and discussion

The responses of the nodal degrees of freedom can be plotted to show that user subroutine *DISP* is providing the same history as the amplitude description.

Input files

udispxxx.inp	Input file for this analysis.
udispxxx.f	User subroutine *DISP* used in udispxxx.inp.
udispxxx_part1.inp	Input file for this analysis with the model defined in terms of an assembly of part instances. This file refers to the user subroutine *DISP* that uses the utility routine `GETPARTINFO`.
udispxxx_part1.f	User subroutine *DISP* used in udispxxx_part1.inp (illustrates the use of the utility routine `GETPARTINFO`).
udispxxx_part2.inp	Input file for this analysis with the model defined in terms of an assembly of part instances. This file refers to the user subroutine *DISP* that uses the utility routine `GETINTERNAL`.

udispxxx_part2.f	User subroutine *DISP* used in udispxxx_part2.inp (illustrates the use of the utility routine `GETINTERNAL`).

DLOAD

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Features tested

User subroutine to specify nonuniform distributed loads.

General comments

User subroutine *DLOAD* can be used to define several types of nonuniform distributed loads. Examples of the use of this subroutine are shown in some of the tests described in other sections of this guide. The use of the subroutine is not limited to the applications shown in these tests. *Pure bending of a cylinder: CAXA elements* and *Cylinder subjected to asymmetric pressure loads: CAXA elements* illustrate the use of subroutine *DLOAD* to apply asymmetric loads to CAXA asymmetric-axisymmetric continuum elements. Refer to the problem description in *Patch test for axisymmetric elements* for an example of the use of the subroutine to define a nonuniform body force in the patch test for axisymmetric continuum stress-displacement elements. Subroutine *DLOAD* is also used in the test described in *Nonuniform crack-face loading and J-integrals* to apply uniform and nonuniform loads to a crack face.

FRIC

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

User subroutine tested in a stress/displacement analysis

Elements tested

B31

Features tested

User subroutine to define frictional behavior for contact surfaces in a stress/displacement analysis.

Problem description

Abaqus provides a Coulomb friction model as the default behavior for frictional interfaces. In this test an alternative constitutive model is used. Here, the interface is assumed to have a viscoplastic behavior so that the slip strain rate is proportional to the shear stress. For this particular example

$$\dot{\gamma} = k\tau,$$

where k=0.001.

A fairly stiff beam element is used to model a rod. The contact between the bottom end of the rod and a three-dimensional rigid surface is modeled by specifying a main-secondary contact pair. The bottom end of the rod constitutes the node-based secondary surface, and the rigid surface represents the main surface. The rigid surface is kept fixed in space throughout the analysis and corresponds to the x–y plane. This configuration is shown in *Figure 1*. The rod, which is perpendicular to the rigid surface (that is, parallel to the z-axis), is forced into contact with the rigid surface and kept in compression by applying a concentrated load in the axial direction at the top of the rod. Subsequently, the rod is forced to slide around the surface by applying a concentrated load vector of the form

$$\mathbf{f}_T = f_x\mathbf{i} + f_y\mathbf{j} + 0\mathbf{k}$$

to the node at the top of the beam element. All rotations are constrained on this node as well.

The first two steps of the analysis set up an equilibrium solution in which the beam element is compressed by a force of 100. The rod is then slid in three steps (Steps 3–5), and each of the steps has a total time of unity. A tangential force $\mathbf{f}_T$ of norm 100 is applied instantaneously during each of these steps to keep the norm of the shear stress vector constant. During these three steps the incremental slip vector and the interfacial shear stresses are checked for consistency with the assumed constitutive law.

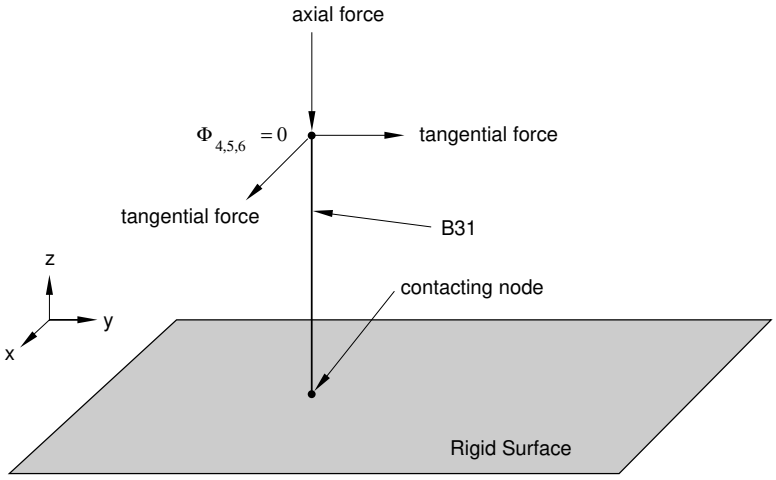


Figure 1: Schematic of first test model for user subroutine `FRIC`.

Reference solution

STEP 3:

A constant tangential force f_x=100 and f_y=0 is applied. The total slip at the end of this step is 0.1 along the *x*-axis since the applied shear stress is held constant with a value of 100 along this axis.

STEP 4:

A constant tangential force f_x=f_y=70.71 is applied. The total slip at the end of this step is .17071 in the *x*-direction and .07071 in the *y*-direction since the applied shear stress is held constant with a value of 70.71 in each direction.

STEP 5:

A constant tangential force f_x=0 and f_y=100 is applied. The total slip at the end of this step is .17071 in each direction since the applied shear stress is held constant with a value of 100 along the *y*-axis.

Results and discussion

The results match the analytical solution for Steps 3, 4, and 5.

Input files

ufricxxx.inp	Stress/displacement analysis.
ufricxxx.f	User subroutine *FRIC* used in ufricxxx.inp.

User subroutine tested in a coupled temperature-displacement analysis

Elements tested

C3D8T

Features tested

User subroutine to define frictional behavior for contact surfaces in a coupled temperature-displacement analysis.

Problem description

In this test the contact interface is assumed to have viscoplastic behavior so that the slip strain rate is proportional to the shear stress and average temperature of the interface. For this particular example

$$\dot{\gamma} = k\tau,$$

where $k = 0.001 + 0.00001\frac{(\theta_1+\theta_2)}{2}$, and θ_1 and θ_2 represent the current temperature of the secondary and main surface nodes, respectively.

Contact is defined between two solid blocks, A and B, as shown in *Figure 2*.

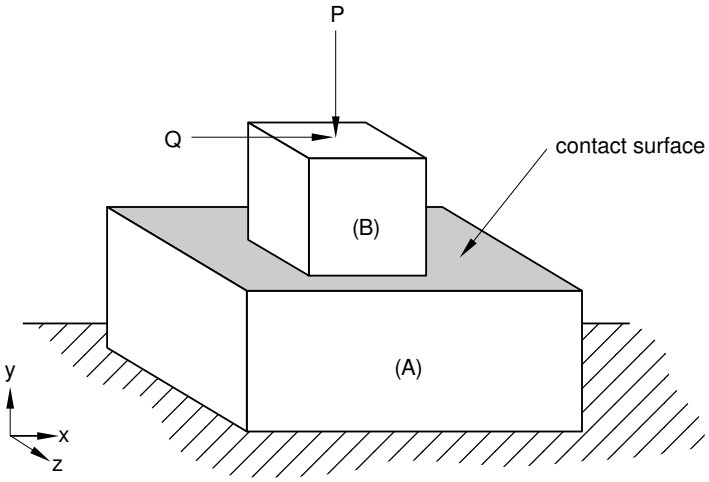


Figure 2: Schematic of second test model for user subroutine FRIC.

The base of block A is fixed in space. The analysis consists of a sequence of steps that are designed to verify the contact conditions and the frictional heat generated due to user-defined friction conditions. The material properties and the different boundary conditions and loads are chosen such that the analytical solution can be easily derived.

In Step 1 contact is established between blocks A and B.

In Step 2 contact between the two blocks is maintained by applying a downward force P=16000 on the top surface of block B. During these two steps the temperature of each block is kept at 0°.

Step 3 verifies that the friction law is applied correctly and that the proper amount of heat is generated due to friction. Block B is slid over block A by instantaneously applying a shear load Q of 100 in the x-direction. The temperature of block B is increased from 0° to 200° while maintaining the temperature of block A at 0°. During the sliding process the top surface of block A is fixed to keep the contact surfaces orthogonal to the y-axis. It is assumed that 50% of the frictional work is transformed into heat and that 50% of that heat goes through each contact surface. During this third step the incremental slip vector, the interfacial shear stresses, and the heat generated are checked for consistency with the assumed constitutive law.

Reference solution

At the end of the third step the total slip can be obtained by integrating the slip rate $\dot{\gamma}$ as

$$\gamma = \int_0^1 k\tau dt = 0.150.$$

In Abaqus this integration is not carried out in a continuous fashion. It is carried out by discretizing the total time in given intervals, leading to the form

$$\gamma = \sum_{i=1}^{n} k\left(i\right)\tau\Delta t\left(i\right),$$

which results in a total slip of 0.155 if the unit time is divided into 10 equal intervals.

The heat generated by friction in each interval is

$$q_g = \eta\tau\frac{\Delta\gamma(i)}{\Delta t(i)} = \eta k\left(i\right)\tau^2,$$

where η=0.5 and τ=100. Half of this quantity goes through each contacting surface.

Results and discussion

Step 3 is carried out over a unit time period in 10 equal increments. As a result, a total slip of 0.155 is obtained. A value closer to 0.150 is obtained when the unit time is divided into more increments. The results obtained at the end of each increment in Step 3 also match the results obtained by analytically summing the slip over each time interval.

Input files

ufricxxy.inp	Coupled temperature-displacement analysis.
ufricxxy.f	User subroutine *FRIC* used in ufricxxy.inp.

FRIC_COEF

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

User subroutine tested in a stress/displacement analysis

Elements tested

B21

Features tested

User subroutine to define the friction coefficient between contact surfaces in a stress/displacement analysis.

Problem description

Abaqus provides user subroutine *FRIC_COEF*, in which complex dependencies of a friction coefficient can be defined on slip rate, pressure, temperature, and field variables. This example verifies the capability by considering the contact response for a Coulomb friction law in which the friction coefficient is of the form

$$\mu = \mu_k + (\mu_s - \mu_k)\ \exp\left(-d_c\dot{\gamma}_{eq}\right),$$

where $\dot{\gamma}_{eq}$ is the slip rate, d_c is the decay coefficient, and μ_s and μ_k are the static and dynamic coefficients of friction, respectively. Both the static and dynamic coefficients are functions of contact pressure, p and the average temperature between the two contacting surfaces, $\overline{\theta}$

$$\mu_s = C_s\overline{\theta} + D_s p\ , \qquad \mu_k = C_k\overline{\theta} + D_k p,$$

where C_s, D_s, C_k, and D_k are constants.

The verification test consists of a rod perpendicular to a fixed rigid surface forced into contact with the rigid surface by a concentrated load applied in the axial direction at the top of the rod. Subsequently, prescribed temperatures and displacements are applied to the rod, forcing the rod to slide along the surface. The contact between the bottom end of the rod and the rigid surface is modeled by specifying a main-secondary contact pair. A node-based secondary surface is defined on the bottom end of the rod. This secondary surface has a contact area of unity; hence, the normal force applied on the rod is equal to the contact pressure.

A second identical rod, subjected to the same loading sequence, serves as the reference solution. The friction behavior for this reference model is entered as tabulated data.

Results and discussion

The user subroutine results closely match the reference solution. The small differences between the solutions are the result of the user subroutine describing the friction coefficient as a continuous exponential function of the slip rate, while the reference solution uses discrete data points with linear interpolation between points.

Input files

ufriccoefd.inp	Analysis with default parameters on the **FRICTION* option.

ufriccoefl.inp	Analysis with the LAGRANGE parameter defined on the **FRICTION* option.
ufriccoefs.inp	Analysis with the SLIP TOLERANCE parameter defined on the **FRICTION* option.
ufriccoefe.inp	Analysis with the ELASTIC SLIP parameter defined on the **FRICTION* option.
ufriccoef.f	User subroutine *FRIC_COEF*.

User subroutine tested with solution-dependent state variables

Elements tested

C3D8

Features tested

User subroutine to define friction coefficient that depends on solution-dependent state variables.

Problem description

In this test the friction coefficient is assumed to be a function of a solution-dependent state variable that is a linear function of equivalent contact slip. A saturation type dependence of the friction coefficient on the solution-dependent variable is assumed:

$$\mu = \mu_I + (\mu_f - \mu_I)(1 - \exp(s)),$$

where the friction coefficient μ varies between an initial value of μ_I and an asymptotic value of μ_f. In the simplest case the solution-dependent variable s is chosen as a linear homogeneous function of equivalent contact slip γ_{eq}; that is, $s = \alpha\gamma_{eq} + \beta$ with parameter $\beta = 0$. The parameter value $\alpha = -0.2$ is chosen.

The test consists of a block meshed with a single element sliding on another block with a single row of elements. The slider block is chosen for the secondary surface, while the main surface is chosen on the other block. Contact is established in the first step. In the second step the slider block is moved with specified displacements. The value of equivalent contact slip is accessed by a call to the utility routine GETVRC, which returns the converged value at the previous increment. The friction coefficient is updated as the slider moves, but its dependence on the solution-dependent variable and, thereby, the equivalent slip is explicit since it corresponds to the converged value for the previous increment. In the third step, which is a dummy step, the slider is held in place; but we allow the solution-dependent variable to evolve to its final value based on the final values of equivalent contact slip. The first three steps use the static procedure. Additional linear steps for steady-state dynamics and frequency extraction are added to verify the user subroutine with other procedure types.

Results and discussion

A separate solution-dependent variable, in this case SDV1, is chosen to store the friction coefficient for visualization purposes only. The friction coefficient evolution can be visualized by contouring SDV1 and can be seen to evolve depending on the value of the solution-dependent variable SDV2 as the slider block moves in the second step.

Input files

ei308fcf_surf_friccoef.inp	Analysis with friction coefficient dependence on solution-dependent state variables.
ei308fcf_surf_friccoef.f	User subroutine *FRIC_COEF* with friction coefficient evolution based on solution-dependent state variables.

User subroutine tested with slip variables at current time

Elements tested

C3D8R

Features tested

User subroutine to define friction coefficient that depends on slip variables at current time.

Problem description

This test incorporates anisotropic friction by making the friction coefficient in each slip direction depend on the slip component in that direction. A bilinear dependence is assumed where the friction coefficient varies linearly with the slip component magnitude until a slip tolerance is reached and remains constant thereafter:

$$\mu_I = a_I\left(\frac{|\gamma_I|}{\gamma_I^{Tol}}\right), \quad |\gamma_I| \le \gamma_I^{Tol}$$

$$\mu_I = \mu_I^{\max}, \qquad |\gamma_I| > \gamma_I^{Tol}$$

where $I = 1, 2$ are indices for the two local tangent directions. For simplicity, identical values of slip tolerance $\gamma_I^{Tol} = 0.1$, coefficients $a_I = 0.3$, and peak friction coefficients $\mu_I^{\max} = 0.3$ are chosen for both directions. The friction behavior is still anisotropic since the slip components might not be equal as the solution evolves.

The derivative of the friction coefficient is a step function at $\gamma_I = 0$, jumping between values $+a_I/\gamma_I^{tol}$ and $-a_I/\gamma_I^{tol}$. In this example the step function for the derivative is replaced by a linear variation with a finite slope and with a zero value at $\gamma_I = 0$ to help convergence.

The test consists of a block meshed with a single element sliding on another block. The slider block is chosen for the secondary surface, while the main surface is chosen on the fixed block. Contact is established in the first step. In the second step the slider block is moved with specified displacements to trace a slip path along a line and returning along the same path. The current values of the slip components are passed via an argument into the user subroutine.

A variant of the user subroutine with utility routine `GETVRC` to retrieve the slip components is also provided. In this case the slip components denote converged values from the previous increment. Since friction coefficient values do not depend implicitly on the slip component values at the current time, the derivatives of the friction coefficient with respect to the slip values are zero.

Results and discussion

The friction coefficients vary linearly starting from zero as a function of the slip component magnitude in the beginning. They reach a specified peak value before decreasing to a value close to zero at the end of the analysis as the slip components become nearly zero again. The shear stresses similarly become small at the end of the analysis.

Using the variant user subroutine with `GETVRC`, the slip components lag by one increment. The resulting friction coefficients do not become small at the end of the analysis because the increment size of the last increment is not small. Therefore, the shear stresses remain high at the end of the analysis, resulting in a poor approximation.

Input files

cslip_cp_ntos_fric_coef_slipval.inp	Analysis with friction coefficient dependence on slip component values specified through the user subroutine.
ucslip_cp_ntos_fric_coef_slipval.f	User subroutine *FRIC_COEF* with friction coefficient evolution based on current slip component values.
ucslip_cp_ntos_fric_coef_slipval_getvrc.f	User subroutine *FRIC_COEF* with friction coefficient evolution based on previous converged slip component values.

GAPCON

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

CPEG4T C3D8T DC3D8 DCC3D8

Features tested

User subroutine to define gap conductance for interface elements that allow for heat transfer.

Problem description

To verify user subroutine *GAPCON*, the thermal interface properties verification tests of *Gap conductance* are repeated using the user subroutine to specify gap conductance values. Results match for both methods of specifying gap conductance values. The tests are set up as cases of uniform one-dimensional heat flux using two- and three-dimensional elements. For the three-dimensional analyses, the temperature results are identical for all nodes located on a particular plane along the direction of heat flow. These include nodes 1–7 at plane *A*, nodes 101–107 at plane *B*, and nodes 201–207 at plane *C*. A steady-state heat transfer analysis is performed in several increments in all cases. Particular values (gap clearance, predefined field variables, etc.) vary during the solution, which in turn influence the conductivity across the interface and, thus, the solution. These values are passed into user subroutine *GAPCON* where an appropriate value of gap conduction is specified, thus affecting the temperature solution.

Results and discussion

The results match the exact solutions.

Input files

ugapconcg.inp	Clearance-dependent conductivity with CPEG4T elements.
ugapconcs.inp	Clearance-dependent conductivity with C3D8T elements.
ugapconcs.f	User subroutine *GAPCON* used in ugapconcg.inp and ugapconcs.inp.
ugapconfs.inp	Field-variable-dependent conductivity with DC3D8 elements.
ugapconfs_gc.inp	Field-variable-dependent conductivity with DC3D8 elements and general contact.
ugapconfs.f	User subroutine *GAPCON* used in ugapconfs.inp and ugapconfs_gc.inp.
ugapconms.inp	Mass-flow-rate-dependent conductivity with DCC3D8 elements.
ugapconms_gc.inp	Mass-flow-rate-dependent conductivity with DCC3D8 elements and general contact.
ugapconms.f	User subroutine *GAPCON* used in ugapconms.inp and ugapconms_gc.inp.
ugapconpg.inp	Pressure-dependent conductivity with CPEG4T elements.
ugapconps.inp	Pressure-dependent conductivity with C3D8T elements.

ugapconps.f	User subroutine *GAPCON* used in `ugapconpg.inp` and `ugapconps.inp`.
ugapconts.inp	Temperature-dependent conductivity with DC3D8 elements.
ugapconts.f	User subroutine *GAPCON* used in `ugapconts.inp`.

GAPELECTR

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

DC2D4E

Features tested

User subroutine to define electrical conductance between surfaces in a coupled thermal-electrical simulation.

Problem description

In coupled thermal-electrical analyses user subroutine *GAPELECTR* is used to define the electrical conductance between two surfaces as a function of their temperatures, the gap between them, and any field variables. The problem definition is the same as in the verification file *ei22vsjc.inp* (*Coupled thermal-electrical surface interaction*), where the gap electrical conductance is defined as a function of the average temperature.

Results and discussion

The results match the verified results of *ei22vsjc.inp*.

Input files

ugapelectr.inp	Input file for this analysis.
ugapelectr_gc.inp	Input file with general contact.
ugapelectr.f	User subroutine *GAPELECTR* used in `ugapelectr.inp` and `ugapelectr_gc.inp`.

HARDINI

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

M3D4

Features tested

User subroutine to define initial conditions for hardening variables $\bar{\varepsilon}^{pl}$ and $\boldsymbol{\alpha}$.

Problem description

User subroutine *HARDINI* is used to define initial conditions for the hardening variables $\bar{\varepsilon}^{pl}$ and $\boldsymbol{\alpha}$. These initial conditions are used to solve for the mechanical response of an M3D4 element with rebar in uniaxial tension. The problem definition is the same as in the verification file *mplchi3nt1.inp* (*Rate-independent plasticity*), where the initial conditions are specified. This input file verifies that initial conditions are assigned correctly using user subroutine *HARDINI*.

Results and discussion

The results match the verified results of *mplchi3nt1.inp*.

Input files

uhardini.inp	Input file for this analysis.
uhardini.f	User subroutine *HARDINI* used in uhardini.inp.

HETVAL

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

DC2D4

Features tested

User subroutine to provide internal heat generation in heat transfer analyses.

Problem description

A two-dimensional rectangular block of material, 10×2, has heat generated within its volume by user subroutine *HETVAL*. The value of the generated heat flux is $r = 0.40483$. The material has specific heat, $c = 0.1431$, and density, $\rho = 0.2829$. A transient thermal analysis with all edges of the volume insulated should give a temperature rate of

$$\frac{d\theta}{dt} = \frac{r}{\rho c} = 10.$$

Results and discussion

Time is incremented by 5 units in each increment of the analysis. From the equation above, therefore, nodal temperatures should increment by 50 units during each increment.

Increment	Time	Temperature
1	5	50
2	10	100
3	15	150
4	20	200
5	25	250
6	30	300
7	35	350
8	40	400
9	45	450
10	50	500

Input files

uhetvalx.inp	Input file for this analysis.
uhetvalx.f	User subroutine *HETVAL* used in uhetvalx.inp.

RSURFU

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

CAX4 S4R

Features tested

User subroutine to define rigid surface geometry.

Problem description

ursurfux.inp

This test verifies that user subroutine *RSURFU* properly generates a three-dimensional rigid surface. The problem consists of forming an elastic sheet around a rigid cylinder. This problem will be compared to the test shown in *Finite-sliding contact between a deformable body and a meshed rigid surface*, which performs the identical analysis using a Bézier surface instead of a user-defined rigid surface.

The cylinder has a radius of 5 inches, and its displacements and rotations are restrained. The sheet has dimensions of 5 inches by 10 inches and is modeled with fifty 4-node S4R shell elements. It is assumed to be elastic with Young's modulus of 3×10^6 lb/in^2 and Poisson's ratio of 0.3. The sheet is initially positioned tangent to the surface of the cylinder, with one edge fixed to the surface of the cylinder. A pressure load of 700 lb/in^2 is applied to the surface of the sheet to form it around the circumference of the cylinder. All of the shell nodes are put into a contact node set with the exception of the nodes along the built-in edge. The contact node set is defined as the secondary surface, and the user-defined rigid surface is defined as the main surface. No frictional behavior is included.

ursurfu2.inp

This test compares two models, one using an analytical rigid surface and the other using user subroutine *RSURFU*. A circular plate of radius 10 and thickness 1 is modeled using two-dimensional, axisymmetric CAX4 elements. The plate is assumed to be elastic with a Young's modulus of 3×10^5 and Poisson's ratio of 0.3. The displacements at the boundary of the plate are restrained. An axisymmetric rigid punch rests on one side of the plate. A load of 1.5×10^5 is applied to the punch to deform the plate.

ursurfu3.inp

This test compares two models, one using an analytical rigid surface and the other using user subroutine *RSURFU*. A 10×10 mesh of S4R shell elements is used to model a square plate. The displacements at the boundary of the plate are restrained. A rigid punch rests on one side of the plate. A load of 3×10^4 is applied to the punch to deform the plate.

Results and discussion

The displacements of the deformed sheet in the first test are within 1% of the results from the Bézier rigid surface verification problem.

In the second and third tests the results of the models using the user subroutines are identical to those of the corresponding models with analytical rigid surfaces.

Input files

ursurfux.inp	Three-dimensional rigid surface compared to Bézier rigid surface.
ursurfux.f	User subroutine *RSURFU* used in ursurfux.inp.
ursurfu2.inp	Two-dimensional axisymmetric rigid surface compared to analytical rigid surface.
ursurfu2_surf.inp	Two-dimensional axisymmetric rigid surface compared to analytical rigid surface using surface-to-surface contact formulation.
ursurfu2.f	User subroutine *RSURFU* to be used with ursurfu2.inp.
ursurfu2_surf.f	User subroutine *RSURFU* to be used with ursurfu2_surf.inp.
ursurfu3.inp	Three-dimensional rigid surface compared to analytical rigid surface.
ursurfu3_surf.inp	Three-dimensional rigid surface compared to analytical rigid surface using the surface-to-surface contact formulation.
ursurfu3.f	User subroutine *RSURFU* to be used with ursurfu3.inp.
ursurfu3_surf.f	User subroutine *RSURFU* to be used with ursurfu3_surf.inp.

SDVINI

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

B31 CPS4 GKPS4

Features tested

User subroutine to define initial conditions for solution-dependent state variables.

Problem description

Two test cases are included. The purpose of both tests is to show that the initial conditions of the solution-dependent state variables are interpreted correctly and to show that the solution-dependent state variables can be updated in another user subroutine.

In the first test case six solution-dependent state variables are initialized in user subroutine *SDVINI* and are subsequently updated in user subroutine *UMAT*. The problem is a trivial linear elastic, static analysis of a single plane stress element. The analysis is repeated for a plane stress gasket element, and identical results are obtained.

In the second test case two solution-dependent state variables are initialized in user subroutine *SDVINI* and are subsequently updated in user subroutine *FRIC*. The expected solution-dependent state variable settings are confirmed in the step-1, increment-1 call to *FRIC*.

Results and discussion

The solution-dependent state variables defined in *SDVINI* are made available properly in the other user subroutines.

Input files

usdvinix.inp	Input file for the *UMAT* analysis.
usdvinix.f	User subroutine *SDVINI* used in usdvinix.inp.
usdvinifric.inp	Input file for the *FRIC* analysis.
usdvinifric.f	User subroutine *SDVINI* used in usdvinifric.inp.

UAMP

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.



Warning:

User subroutine *UAMP* provides the user with a very general option to interface with the code. With any use of this subroutine interface, extensive verification should be done to make sure that the results are correct.

Features tested

User subroutine *UAMP* to define amplitudes.

Problem description

The finite element models for most test cases consist of simple linear truss or connector elements. User subroutine *UAMP* is used to define amplitudes that are subsequently used to drive certain loading options such as concentrated loads, boundary conditions, and connector motions. In most cases, the *UAMP* user-defined amplitudes are simple linear ramps. The results from the analyses are compared against reference results obtained using identical models with equivalent tabular amplitude definitions.

User subroutine *UAMP* can make use of sensor definitions and of state variables, and a number of tests exercise these features. In certain tests (such as when a user-defined amplitude is used to drive the displacement boundary condition) the user subroutine may compute derivatives, integrals, and second derivatives of the amplitude function being defined.

Results and discussion

The verification consists of comparing the results obtained from the model using user-defined amplitudes with the corresponding model using tabular amplitudes. The results match very well, as expected.

Input files

uamp_ramp_simple_cload.inp	Concentrated load force scaled by an amplitude defined in user subroutine *UAMP* in uamp_ramp_simple.f.
uamp_ramp_simple_bcdisp.inp	Displacement boundary condition scaled by an amplitude defined in user subroutine *UAMP* in uamp_ramp_simple.f.
uamp_ramp_simple_restart.inp	Displacement boundary condition in a restart analysis scaled by an amplitude defined in user subroutine *UAMP* in uamp_ramp_simple.f.
uamp_ramp_simple_bcvel.inp	Velocity boundary condition scaled by an amplitude defined in user subroutine *UAMP* in uamp_ramp_simple.f.
uamp_ramp_simple_bcacc.inp	Acceleration boundary condition scaled by an amplitude defined in user subroutine *UAMP* in uamp_ramp_simple.f.

uamp_ramp_simple_connmot.inp	Displacement connector motion scaled by an amplitude defined in user subroutine *UAMP* in uamp_ramp_simple.f.
uamp_ramp_simple.f	User subroutine *UAMP* defining a simple ramp amplitude using state variables.
uamp_ramp_state_cload.inp	Concentrated load force scaled by an amplitude defined in user subroutine *UAMP* in uamp_ramp_state.f.
uamp_ramp_state_connmot.inp	Displacement connector motion scaled by an amplitude defined in user subroutine *UAMP* in uamp_ramp_state.f.
uamp_ramp_state.f	User subroutine *UAMP* defining a simple ramp amplitude using state variables.
uamp_deriv_bcdisp.inp	Displacement connector motion scaled by an amplitude defined in user subroutine *UAMP* in uamp_deriv.f.
uamp_deriv.f	User subroutine *UAMP* defining cubic function amplitude, which also computes derivatives.
uamp_ramp_sensor_connmot.inp	Displacement connector motion scaled by an amplitude defined in user subroutine *UAMP* in uamp_ramp_sensor.f.
uamp_ramp_sensor.f	User subroutine *UAMP* defining a simple ramp amplitude using sensors.
uamp_manysensors.inp	Displacement connector motion scaled by an amplitude defined in user subroutine *UAMP* in uamp_manysensors.f.
uamp_manysensors.f	User subroutine *UAMP* defining a simple ramp amplitude using sensors.

UANISOHYPER_INV and VUANISOHYPER_INV

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Features tested

Family of user subroutines to define anisotropic hyperelastic material behavior.

Elements tested

C3D8 C3D8H C3D8R CPEG4 CPE4H CPE4R CPE8R

Problem description

This set of verification problems is primarily intended to test the variables that are passed into *UANISOHYPER_INV* in Abaqus/Standard or *VUANISOHYPER_INV* in Abaqus/Explicit. These tests also verify that the derivatives of the strain energy function defined by the user are transferred properly to the solution process. In each test the material properties are specified using the user strain energy potential for the testing elements, for which the strain energy function and the associated derivatives are defined in user subroutines *UANISOHYPER_INV* and *VUANISOHYPER_INV*. Each test contains one reference element with material properties specified with anisotropic hyperelasticity, which provides the reference solution. Three different sets of material data are used, as described below.

Material 1: Holzapfel-Gasser-Ogden material with two families of fibers:

Holzapfel-Gasser-Ogden coefficients:

C_{10} = 7.64., k_1 = 996.6, k_2 = 524.6, κ = 0.226.

Fiber directions (N=2):

$$\mathbf{A}_1 = (\cos\gamma, \sin\gamma, 0)\,,$$

$$\mathbf{A}_2 = (\cos\gamma, -\sin\gamma, 0)\,,$$

with $\gamma = 49.98^o$.

Compressible case: $D = 10^{-8}$

Material 2: Polynomial (N=2) isotropic hyperelastic behavior

Polynomial coefficients (N=2):

C_{10} = 100.0

C_{01} = 50.0

C_{20} = 10.0

C_{11} = 20.0

C_{02} = 30.0

Compressible case: D_1 = 0.01, D_2=0.0

Material 3: Generalized Fung energy function implemented in terms of pseudo invariants. Two implementations are considered: one with the components of the modified Green strain expressed in terms of $\overline{I}_{4(\alpha\beta)}$ type invariants, and the other in terms of $\overline{I}_{4(\alpha\beta)}$ and $\overline{I}_{5(\alpha\beta)}$ type invariants.

Fung coefficients:

$c = 26.95 \times 10^3$

b_{1111} = 0.9925

b_{1122} = 0.0749

b_{2222} = 0.4180

b_{1133} = 0.0295

b_{2233} = 0.0193

b_{3333} = 0.0089

b_{1212} = 5.0

b_{1313} = 5.0

b_{2323} = 5.0

Compressible case: D = 0.1

Results and discussion

The tests in this section are set up as cases of homogeneous deformation of a single element of unit dimensions. Consequently, the results are identical for all integration points within the element. In each case the results in the testing elements match the solution in the reference element.

Input files

Abaqus/Standard input files

uaniso_inv_hgople.inp	Holzapfel-Gasser-Ogden anisotropic hyperelasticity, compressible, uniaxial plane strain tension.
uaniso_inv_isople.inp	Polynomial hyperelasticity, incompressible, uniaxial plane strain tension, hybrid elements.
uaniso_inv_fung.inp	Fung anisotropic hyperelasticity, compressible, uniaxial plane strain tension.
uanisohyper_inv.f	User subroutine *UANISOHYPER_INV* used in the above tests.

Abaqus/Explicit input files

vuaniso_inv_hgople.inp	Holzapfel-Gasser-Ogden anisotropic hyperelasticity, compressible, uniaxial plane strain tension.
vuaniso_inv_isople.inp	Polynomial hyperelasticity, compressible, uniaxial plane strain tension, hybrid elements.
vuaniso_inv_fung.inp	Fung anisotropic hyperelasticity, compressible, uniaxial plane strain tension.
vuanisohyper_inv.f	User subroutine *VUANISOHYPER_INV* used in the above tests.

UCOHESIVEOFFSET

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Features tested

User subroutine *UCOHESIVEOFFSET* to define cohesive offset.

Problem description

These tests are related to hydaulic-driven crack propagation. The finite element model for each test case consists of two steps. In the first step, the fluid is injected into the crack mouth so that the crack will propagate and its opening will increase. In the second step, injection of the fluid is stopped. Due to fluid leakoff to the surrounding media, the crack will close. Using user subroutine *UCOHESIVEOFFSET*, we can hold the crack opening to either a constant offset height or its maximum opening height as achieved in the previous step.

Results and discussion

The results match the expected results well.

Input files

ucohesiveoffset_1.inp	Constant cohesive offset defined in user subroutine *UCOHESIVEOFFSET* in `ucohesiveoffset_1.f` with `jflgopt`=1.
ucohesiveoffset_1.f	Constant cohesive offset defined with `jflgopt`=1.
ucohesiveoffset_2.inp	Cohesive offset defined in user subroutine *UCOHESIVEOFFSET* as the maximum opening in `ucohesiveoffset_2.f` with `jflgopt`=1.
ucohesiveoffset_2.f	Cohesive offset defined with `jflgopt`=1.
ucohesiveoffset_3.inp	Constant cohesive offset defined in user subroutine *UCOHESIVEOFFSET* in `ucohesiveoffset_3.f` with `jflgopt`=2.
ucohesiveoffset_3.f	Constant cohesive offset defined with `jflgopt`=2.
ucohesiveoffset_4.inp	Cohesive offset defined in user subroutine *UCOHESIVEOFFSET* as the maximum opening in `ucohesiveoffset_4.f` with `jflgopt`=2.
ucohesiveoffset_4.f	Cohesive offset defined with `jflgopt`=2.

Figures

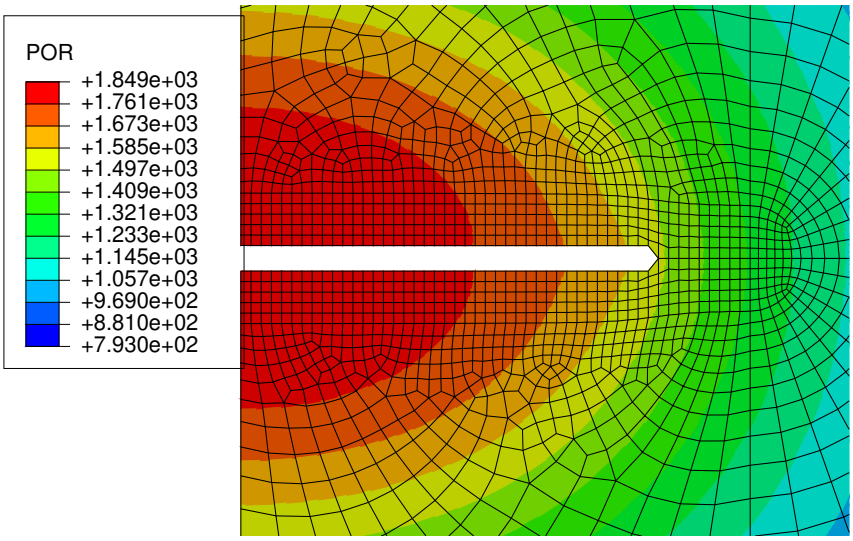


Figure 1: The final crack profile for `uucohesiveoffset_1.inp`.

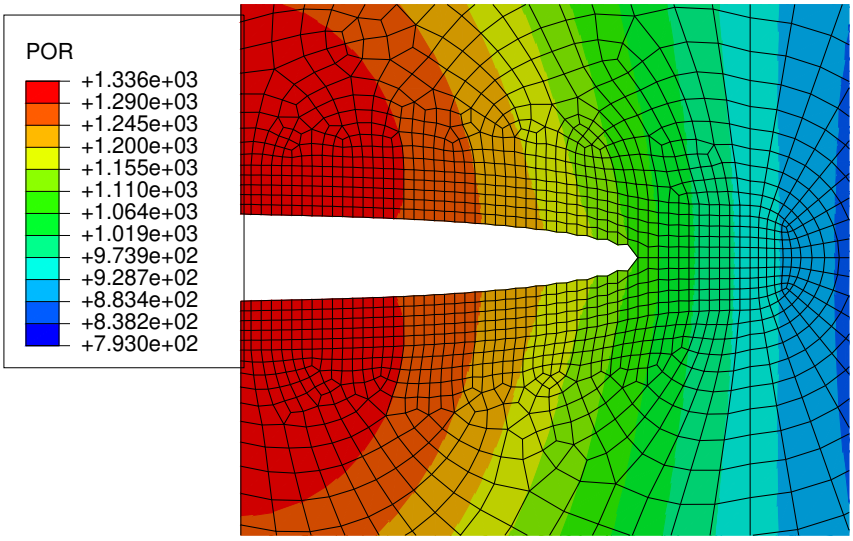


Figure 2: The final crack profile for `uucohesiveoffset_2.inp`.

UCREEPNETWORK

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D8H

Features tested

User subroutine to define time-dependent behavior for models defined within the parallel rheological framework.

Problem description

The input files consist of one element each with the material response specified using the parallel rheological framework with two or three viscous networks. The Bergstrom-Boyce creep model is defined in network one and network three (for the model with three viscous networks) using user subroutine *UCREEPNETWORK*. A two-step analysis is performed. The specimen is preloaded in the first static step in which viscous effects are not active. In the second, quasi-static, step the viscous effects are activated and a constant load is maintained throughout the step, so the material is subjected to creep. The results were compared to the results obtained with an equivalent model that uses the built-in Abaqus Bergstrom-Boyce model.

Results and discussion

The results obtained using user subroutine *UCREEPNETWORK* and the built-in Abaqus models are identical.

Input files

viscnet_c3d8h_ab_bb_n2_user.inp	The model with two viscous networks that uses the user subroutine to define the Bergstrom-Boyce creep model.
viscnet_c3d8h_ab_bb_n3_user.inp	The model with three viscous networks that uses the user subroutine to define the Bergstrom-Boyce creep model.
viscnet_bb_user.f	User subroutine to define the Bergstrom-Boyce creep model.

UCURE

Products: Abaqus/Standard

References:

- *Curing Processes in Polymers*

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D8RT CAX4RT CPE4RT T2D2T
DC2D4 DC2D4E DC3D8E

Problem description

These tests verify the correctness of the cure modeling capabilities in Abaqus/Standard when the cure kinetics are defined in user subroutine *UCURE*. The Kamal and tabular cure kinetics models are implemented in the user subroutine, and the results are compared with the results obtained using equivalent native Abaqus models.

All the tests consist of one element. No stresses or thermal loads are applied, and sufficient boundary conditions are prescribed to prevent rigid body motion. In all cases, an initial temperature of 75 is prescribed. To begin the curing process, initial conditions are used to assign small initial values of degree of cure equal to 0.001.

Material: The material parameters used are given below. The elastic constants and the time-temperature shift function are defined as a function of the degree of cure using a field variable associated with output variable DOC.

Elasticity

Table 1: Linear elasticity.

E	ν	Temperature	Field variable (DOC)
0.003	0.45	20.0	0.0
19.2	0.45	200.0	1.0

Viscoelasticity

Table 2: Prony series coefficients.

g_1	k_1	τ_1
0.4	0.0	2×10^{-7}
0.3	0.0	4.0
0.2	0.0	5×10^{8}

Table 3: Tabular definition of the shift function.

Logarithm of the shift function	Temperature	Field variable (DOC)
-3.5	20.0	0.0
-3.0	200.0	0.0
0.5	20.0	1.0
-16.0	200.0	1.0

Thermal properties

Table 4: Conductivity and specific heat.

k	c
3.2×10^{-4}	1.86×10^{6}

Reaction kinetics

Table 5: Kamal model.

Z_1	E_1	m_1	n_1	b_1
3.5×10^{14}	10^5	1.05	1.3	0.0

The values for the tabular model were generated from the Kamal's model described in the table above.

Shrinkage coefficients

Table 6: Volumetric and isotropic shrinkage coefficients.

Isotropic shrinkage coefficient, γ	Volumetric shrinkage coefficient, γ_{vol}
0.01433	0.043

Table 7: Orthotropic shrinkage coefficients.

γ_{11}	γ_{22}	γ_{33}
0.01433	0.01	0.007

Table 8: Anisotropic shrinkage coefficients.

γ_{11}	γ_{22}	γ_{33}	γ_{12}	γ_{13}	γ_{23}
0.01433	0.01	0.007	0.002	0.001	0.0015

Volumetric heat generation parameters

Table 9: Specific heat of reaction and initial density.

Q	ρ_0
2.5×10^7	1.25×10^{-9}

Results and discussion

The results obtained using cure kinetics models implemented in user subroutine *UCURE* agree very well with the results obtained using native Abaqus models.

Input files

ucure_Kamal_iso_cax4rt.inp	Kamal reaction kinetics model implemented in user subroutine *UCURE*, isotropic cure shrinkage strain, and CAX4RT element.
ucure_Kamal_iso_cpe4rt.inp	Kamal reaction kinetics model implemented in user subroutine *UCURE*, isotropic cure shrinkage strain, and CPE4RT element.
ucure_tabular_iso_cpe4rt.inp	Reaction kinetics model specified in tabular form in user subroutine *UCURE*, isotropic cure shrinkage strain, and CPE4RT element.
ucure_Kamal_iso_c3d8rt.inp	Kamal reaction kinetics model implemented in user subroutine *UCURE*, isotropic cure shrinkage strain, C3D8RT element, and maximum conversion specified.
ucure_Kamal_iso_max_conversion_cpe4rt.inp	Kamal reaction kinetics model implemented in user subroutine *UCURE*, isotropic cure shrinkage strain, CPE4RT element, and maximum conversion specified.
ucure_Kamal_iso_t2d2t.inp	Kamal reaction kinetics model implemented in user subroutine *UCURE*, isotropic cure shrinkage strain, and T2D2T element.
ucure_Kamal_dc2d4_HT.inp	Kamal reaction kinetics model implemented in user subroutine *UCURE*, transient heat transfer analysis, and DC2D4 element.
ucure_Kamal_max_conversion_dc2d4_HT.inp	Kamal reaction kinetics model implemented in user subroutine *UCURE*, transient heat transfer analysis, DC2D4 element, and maximum conversion specified.
ucure_tabular_dc2d4_HT.inp	Reaction kinetics model specified in tabular form in user subroutine *UCURE*, transient heat transfer analysis, and DC2D4 element.
ucure_Kamal_dc3d8e_CTE.inp	Kamal reaction kinetics model implemented in user subroutine *UCURE*, transient coupled thermal-electrical analysis, and DC3D8E element.
ucure_tabular_dc2d4e_CTE.inp	Reaction kinetics model specified in tabular form in user subroutine *UCURE*, transient coupled thermal-electrical analysis, and DC2D4E element.
ucure_Kamal.f	User subroutine *UCURE* used in `ucure_Kamal_iso_cax4rt.inp`, `ucure_Kamal_iso_cpe4rt.inp`, `ucure_Kamal_iso_c3d8rt.inp`, `ucure_Kamal_iso_max_conversion_cpe4rt.inp`, `ucure_Kamal_iso_t2d2t.inp`, `ucure_Kamal_dc2d4_HT.inp`, `ucure_Kamal_max_conversion_dc2d4_HT.inp`, and `ucure_Kamal_dc3d8e_CTE.inp`.
ucure_tabular.f	User subroutine *UCURE* used in `ucure_tabular_iso_cpe4rt.inp`, `ucure_tabular_dc2d4_HT.inp`, and `ucure_tabular_dc2d4e_CTE.inp`.

UEL

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.



Warning:

User subroutine *UEL* provides the user with a very general option to interface with the code. With any use of this subroutine interface, extensive verification should be done to make sure that the results are correct.

Structural user element

Elements tested

T3D2

Features tested

User subroutine to define the element mass matrix, element operator matrix, and right-hand side vector.

Problem description

The finite element model for each test case consists of two separate but identical meshes of a simple truss. One mesh consists of five T3D2 elements, and the other consists of five equivalent user-defined elements.

Four test cases are performed as described below.

uellinea.inp

In this problem a linear analysis is run that uses the data line input option to specify the stiffness and mass matrix of the user element. This means that the subroutine is not used, but rather the stiffness and mass matrix for the user element are tested. One end of the truss is constrained. In the first step a load is applied at the second end of the truss. In the second step the eigenfrequencies of the truss are calculated.

uelnonli.inp

The same problem is solved as in uellinea.inp, but the user subroutine is used. The problem is still linear, but there is no assumption of linearity in the user-defined element.

uelriksx.inp

In this job the load is applied gradually using the modified Riks method for proportional loading cases.

ueldynam.inp

In this case the load is applied instantaneously as the implicit dynamics procedure is used to calculate the results for the first 10 increments.

Results and discussion

The verification consists of comparing the results from the T3D2 models to the results obtained from the user element models, since they should be identical.

Input files

uellinea.inp	Linear analysis with data line input option.
uelnonli.inp	Linear analysis with user subroutine.
uelnonli.f	User subroutine *UEL* used in uelnonli.inp.
uelriksx.inp	Analysis with RIKS procedure.
uelriksx.f	User subroutine *UEL* used in uelriksx.inp.
ueldynam.inp	Analysis with implicit dynamics.
ueldynam.f	User subroutine *UEL* used in ueldynam.inp.

Heat transfer user element

Elements tested
DC2D8

Features tested

User subroutine to define the element operator matrix and the right-hand side vector.

Problem description

The finite element model in each test case consists of two separate but identical meshes of a rectangular block. One mesh consists of two DC2D8 elements, and the other consists of two equivalent user-defined elements. The elements in each mesh have an irregular shape to ensure that the interpolation is consistent for the two element types.

Two test cases are performed as described below.

ueltran1.inp

In this problem a transient analysis is performed in which a distributed flux is specified on the left-hand side of the domain and a convection film condition on the right-hand side of the domain. The top and bottom surfaces of the block are adiabatic. The analysis is run until a steady-state condition is satisfied.

ueltran2.inp

The problem outlined in *ueltran1.inp* is solved again, but in this case the thermal conductivity is temperature-dependent. In addition, the unsymmetric equation solver is invoked for user-defined elements with unsymmetric element matrices. For user element operator matrices to be identical to those of the DC2D8 elements, the additional unsymmetric contribution of the temperature-dependent thermal conductivity term must be included (see *Uncoupled heat transfer analysis*).

Results and discussion

The verification consists of comparing the results from the DC2D8 models to the results obtained from the user element models, since they should be identical.

Input files

ueltran1.inp	Transient analysis.
ueltran1.f	User subroutine *UEL* used in ueltran1.inp.
ueltran2.inp	Transient analysis with temperature-dependent thermal conductivity.
ueltran2.f	User subroutine *UEL* used in ueltran2.inp.

Wave utility subroutines

Features tested

User subroutine to test the utility subroutines for fluid kinematic data in Abaqus/Aqua analyses.

Problem description



Warning:

A dummy user element is used to call the utility subroutines for fluid kinematic data. The fluid kinematic data are requested at different points. Three test cases are performed.

Results and discussion

The verification consists of comparing the results returned by the utility subroutines with the results from the wave theory. The results from the wave theory are given in the input files as comment lines.

Input files

uelutwv1.inp	Gets the velocities and accelerations from the utility subroutine `GETWAVEVEL` for a Stokes' wave at a few points.
uelutwv1.f	User subroutine *UEL* used in uelutwv1.inp.
uelutwv2.inp	Gets the velocities and accelerations from the utility subroutine `GETWAVEVEL` for an Airy wave at a few points.
uelutwv2.f	User subroutine *UEL* used in uelutwv2.inp.
uelutwv3.inp	Gets the steady current velocities and wind velocities from the utility subroutines `GETCURRVEL` and `GETWINDVEL`, respectively, for points below and above the still fluid surface.
uelutwv3.f	User subroutine *UEL* used in uelutwv3.inp.

Acoustic user element

Elements tested

AC3D8

Features tested

User subroutine *UEL* to define the element operator left-hand-side matrix and the right-hand-side vector.

Problem description

The model consists of one element. Two test cases are presented. In the first case, the element is implemented through user subroutine *UEL*. The second case is a transient dynamic analysis using the AC3D8 acoustic element.

Results and discussion

The verification consists of comparing the results from the AC3D8 element model to the results obtained from the user element model, since they should be identical.

Input files

acoustuserelemdyn.inp	Transient dynamic analysis with user subroutine *UEL*.
acoustuserelemdyn.f	User subroutine *UEL* used in acoustuserelemdyn.inp.
acoustuserelemdyn2.inp	Transient dynamic analysis using the AC3D8 acoustic element.

UELMAT

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.



Warning:

User subroutine *UELMAT* provides the user with a very general option to interface with the code. With any use of this subroutine interface, extensive verification should be done to make sure that the results are correct.

Structural user element

Elements tested

User element.

Features tested

Accessing various Abaqus materials from a user element material point.

Problem description

The finite element model for each test case consists of a single 4-node user element subjected to uniaxial deformation. The element corresponds to Abaqus element CPE4. Different Abaqus materials are accessed from user subroutine *UELMAT* in each test.

Results and discussion

The results agree with the results obtained using an identical model with Abaqus element CPE4 instead of the user element.

Input files

uelmat_linela_2d.inp	Linearly elastic material.
uelmat_cappla_2d.inp	Modified Drucker-Prager/Cap plasticity model.
uelmat_creep_2d.inp	Material with creep behavior.
uelmat_druckerprager_2d.inp	Drucker-Prager plasticity model.
uelmat_crushfoam_2d.inp	Crushable foam plasticity model.
uelmat_hyperelas_neohook_2d.inp	Neo-Hookean hyperelastic model.
uelmat_hyperfoam_2d.inp	Hyperelastic foam.
uelmat_nkh_2d.inp	Nonlinear kinematic hardening model.
uelmat_pormetalpla_2d.inp	Porous metal plasticity model.
uelmatmech.f	User subroutine *UELMAT* used with all the input files.

Heat transfer user element

Elements tested

User element.

Features tested

User subroutine to define the element operator matrix and the right-hand-side vector.

Problem description

The finite element model for each test case consists of a single 4-node user element. The element corresponds to the Abaqus element DC2D4. The boundary conditions consist of applying heat fluxes to two of the element nodes and applying constant temperatures to the remaining nodes. Steady-state and transient analyses are tested.

Results and discussion

The results agree with the results obtained using an identical model with the Abaqus element DC2D4 instead of the user element.

Input files

uelmat_ht_2d_ss.inp	Steady-state heat transient analysis.
uelmat_ht_2d_trans.inp	Transient heat transient analysis.
uelmatht.f	User subroutine *UELMAT* used with uelmat_ht_2d_ss.inp and uelmat_ht_2d_trans.inp.

UEPACTIVATIONVOL

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D8 DC3D8

Features tested

User subroutine to activate elements during an analysis.

Problem description

The model consists of 10 elements that are activated during the analysis. The activation of elements is defined either at specified step times or at specified increments. This feature is tested with static and heat transfer analysis procedures.

Results and discussion

The elements in all cases were activated at the times and increments specified in the user subroutines.

Input files

elactiv_dc3d8_ht.inp	Heat transfer analysis.
elactiv_dc3d8_ht.f	User subroutine to define activation at specified times for the heat transfer analysis.
elactiv_c3d8_static.inp	Static analysis.
elactiv_c3d8_static.f	User subroutine to define activation at specified increments for the static analysis.

UEXPAN

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

CPE4

Features tested

User subroutine to define increments of thermal strains.

Problem description

User subroutine *UEXPAN* is used to model the thermal expansion behavior of a linear elastic material. The thermal expansion behavior is modeled as isotropic in *uexpan1x.inp*, and it is modeled as orthotropic in *uexpan2x.inp*. The thermal expansion behavior is modeled as a function of field variables in *uexpanfv.inp*.

In all the tests a single CPE4 element with unit dimensions is used in the finite element model. The material properties in these tests are E = 30.0E6 and ν = 0.3.

Results and discussion

The results match the exact solutions.

Input files

uexpan1x.inp	Isotropic thermal expansion behavior.
uexpan1x.f	User subroutine *UEXPAN* used in uexpan1x.inp.
uexpan2x.inp	Orthotropic thermal expansion behavior.
uexpan2x.f	User subroutine *UEXPAN* used in uexpan2x.inp.
uexpanfv.inp	Thermal expansion behavior is a function of field variables.
uexpanfv.f	User subroutine *UEXPAN* used in uexpanfv.inp.

UFLUID

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Features tested

This section provides basic verification tests for the fluid behavior associated with fluid elements that are generated in Abaqus/Standard when the fluid cavity capability is used.

Elements tested

F2D2 SPRING1

Features tested

User subroutine to define fluid density and fluid compliance for an ideal gas.

Problem description

The fluid is modeled using a two-dimensional fluid block that measures 1 × 1 with unit thickness. The user-defined fluid is modeled as an ideal gas with the following properties:

Ambient pressure, p_A = 14.7

Absolute zero temperature, θ^Z = –460.

Reference density, ρ_R = 10.0

Reference pressure for density, p_R = 0.

Reference temperature for density, θ_R = 200.

Loading: The following five steps are executed:

1. Load the fluid to induce a pressure of 100.0 units.
2. Increase fluid temperature to 300.0.
3. Add prescribed amount of fluid.
4. Remove prescribed amount of fluid.
5. Decrease fluid temperature to 200.0.

Results and discussion

The results match the exact solution.

Input files

ufluidxx.inp	Input file for this analysis.
ufluidxx.f	User subroutine *UFLUID* used in ufluidxx.inp.

UGENS

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

S8R SAX2 SAXA22

Features tested

User subroutine to define a shell section stiffness and right-hand side vector for various shell element types. The subroutine argument list is stored in an array reserved for solution-dependent state variables. This array is then written to the results file for verification.

Problem description

To verify user subroutine *UGENS*, the data line input option is used to specify the shell section stiffness and thickness of the shell elements (passed into *UGENS* via array `PROPS`). The general shell section stiffness is determined from a prior analysis along with a material reference from which Abaqus determines equivalent section properties.

ugensbvr.inp

This problem is discussed in detail in *The barrel vault roof problem*. A 4×4 mesh of S8R elements is used to model a deeply arched roof supported only by diaphragms at its curved edges. The first and second steps verify the response to thermal loading as perturbation and general steps, respectively. The coefficient of thermal expansion is taken as 1.0×10^{-6} at a reference temperature of 70°. The structure is heated to 120° from an initial temperature of 70°. The thermal properties, the section force, and the moment vectors, as well as the thermal loading, are all specified in user subroutine *UGENS*. In the third step a frequency extraction is performed to determine the eigenmodes of the structure. In the fourth step the structure is subjected to a body force in the vertical direction while the previously applied thermal loading is removed. The buckling loads are then determined in the fifth step.

ugenscan.inp

SAXA22 elements are used to model a cantilevered pipe loaded at its tip. This problem is discussed in detail in *Cantilever beam analyzed with CAXA and SAXA elements*.

ugenspsx.inp

A mesh of five SAX2 elements is used to model one-half of a hollow sphere subject to a point load applied in the radial direction. This problem is discussed in detail in *The pinched sphere problem*.

Results and discussion

The verification consists of comparing results from the above models to those in a general shell section without user subroutine *UGENS*. In each case the results were identical. The values of the subroutine arguments are verified via the results file.

Input files

ugensbvr.inp	*The barrel vault roof problem*.
ugensbvr.f	User subroutine *UGENS* used in ugensbvr.inp.
ugenscan.inp	*Cantilever beam analyzed with CAXA and SAXA elements*.
ugenscan.f	User subroutine *UGENS* used in ugenscan.inp.
ugenspsx.inp	*The pinched sphere problem*.
ugenspsx.f	User subroutine *UGENS* used in ugenspsx.inp.

UHARD

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D8 C3D8T

Features tested

User subroutine to define isotropic hardening for Mises plasticity, the combined hardening material model, and porous metal plasticity.

Problem description

This set of verification problems tests many of the variables that are passed into *UHARD*, such as material properties, step times, and field variable increment data. These tests also verify that the user-defined quantities of yield stress and its derivatives are transferred properly to the solution process. These tests are modifications of the tests described in *Rate-independent plasticity* and *Rate-dependent plasticity in Abaqus/Standard*. For the problems selected from these sections, wherever an elastic-plastic material was defined, a user-defined hardening has been implemented in place of the corresponding keyword hardening definition. The structure being analyzed is a cube made of a single C3D8 element (or a C3D8T element when a coupled temperature-displacement procedure is tested).

Material:

Elasticity

Young's modulus, E=200.0E3

Poisson's ratio, ν=0.3

Plasticity

Plasticity descriptions match those of the keyword hardening descriptions in the rate-dependent and rate-independent verification problems referenced.

Results and discussion

The tests in this section are set up as cases of homogeneous deformation of a single element of unit dimensions. Consequently, the results are identical for all integration points within the element. In each case the results match those of the referenced problem with the keyword hardening description.

Problems that test adiabatic and coupled temperature-displacement procedures have density, specific heat, and the inelastic heat fraction defined as unity and conductivity defined as zero. The resulting adiabatic temperature rise is confirmed to agree with the approximate solution determined from problems run with equivalent keyword definitions of hardening.

Input files

uhardmts.inp	Mises plasticity, uniaxial tension test. Similar to mpliho3hut.inp.
uhardmts_pormetal.inp	Porous metal plasticity, uniaxial tension test. Similar to mpliho3hut.inp.
uhardmts.f	User subroutine *UHARD* used in uhardmts.inp and uhardmts_pormetal.inp.
uhardmta.inp	Mises plasticity, adiabatic uniaxial tension test. Similar to mpliho3hut.inp.
uhardmta.f	User subroutine *UHARD* used in uhardmta.inp.
uhardmtc.inp	Mises plasticity, coupled temperature-displacement uniaxial tension test. Similar to mpliho3hut.inp.
uhardmtc_pormetal.inp	Porous metal plasticity, coupled temperature-displacement uniaxial tension test. Similar to mpliho3hut.inp.
uhardmtc.f	User subroutine *UHARD* used in uhardmtc.inp and uhardmtc_pormetal.inp.
uhardmss.inp	Mises plasticity, simple shear test. Similar to mpliho3gsh.inp.
uhardmss_pormetal.inp	Porous metal plasticity, simple shear test. Similar to mpliho3gsh.inp.
uhardmss.f	User subroutine *UHARD* used in uhardmss.inp and uhardmss_pormetal.inp.
uhardcts.inp	Mises plasticity with combined hardening, uniaxial tension with temperature and field variable dependence. Similar to mplchb2hut.inp.
uhardcts.f	User subroutine *UHARD* used in uhardcts.inp.
uhardrts.inp	Rate-dependent Mises plasticity, uniaxial tension test, temperature-dependent power law. Similar to mproot3hut.inp.
uhardrts_pormetal.inp	Porous metal plasticity, uniaxial tension test, temperature-dependent power law. Similar to mproot3hut.inp.
uhardrts.f	User subroutine *UHARD* used in uhardrts.inp and uhardrts_pormetal.inp.

UHYPER_STRETCH

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

User subroutine for hyperelastic model

Elements tested

C3D8 C3D8RH CPE4 CPS4 CPS4R

Problem description

This set of verification problems tests the correctness of the results obtained using user subroutine *UHYPER_STRETCH* for various element types and loading paths. In addition, it tests the variables passed into *UHYPER_STRETCH* and the usage of the property table in *UHYPER_STRETCH*.

Each test consists of two coincident elements. Both elements use the same material model and are subject to identical load histories. One of the elements is a reference element with the hyperelastic behavior specified using either a native Abaqus hyperelastic model (neo-Hookean or polynomial [N=2] model) or user subroutine *UHYPER*. The same material response is specified in the second element using user subroutine *UHYPER_STRETCH*. Four different sets of material data are used. All the materials and their coefficients are chosen such that they can be expressed in the Valanis-Landel form (see *Valanis-Landel Form*), which is the required form by user subroutine *UHYPER_STRETCH*.

Material 1:: Neo-Hookean hyperelastic material.

Table 1: Neo-Hookean material coefficients.

C_{10}	D_1
80.0	2.013423×10^{-4}

Material 2:: Polynomial (N=2) isotropic hyperelastic behavior.

Table 2: Polynomial coefficients.

C_{10}	C_{01}	C_{11}	C_{20}	C_{02}	D_1
480.0	-160.0	0.0	80.0	0.0	2.013423×10^{-4}

Material 3:: Polynomial (N=2) isotropic hyperelastic behavior with temperature-dependent coefficients.

Table 3: Temperature-dependent polynomial coefficients.

C_{10}	C_{01}	C_{11}	C_{20}	C_{02}	D_1	Temperature

960.0	-320.0	0.0	160.0	0.0	2.013423×10^{-4}	0.0
480.0	-160.0	0.0	80.0	0.0	2.013423×10^{-4}	20.0
240.0	-80.0	0.0	40.0	0.0	2.013423×10^{-4}	40.0

Material 4:: Polynomial (N=2) isotropic hyperelastic behavior with field variable-dependent coefficients.

Table 4: Field variable-dependent polynomial coefficients.

C_{10}	C_{01}	C_{11}	C_{20}	C_{02}	D_1	Field variable
960.0	-320.0	0.0	160.0	0.0	2.013423×10^{-4}	0.0
480.0	-160.0	0.0	80.0	0.0	2.013423×10^{-4}	15.0
240.0	-80.0	0.0	40.0	0.0	2.013423×10^{-4}	30.0

Results and discussion

The tests in this section are set up as cases of homogeneous deformation. In each case, the results in the testing elements obtained using user subroutine *UHYPER_STRETCH* match the results in the reference element obtained using either user subroutine *UHYPER* or a native Abaqus hyperelastic model.

Input files

uhyper_stretch_c3d8_comp_shear.inp	Neo-Hookean hyperelasticity, compressible, three-dimensional elements, simple shear.
uhyper_stretch_c3d8_comp_tension.inp	Neo-Hookean hyperelasticity, compressible, three-dimensional elements, uniaxial tension.
uhyper_stretch_c3d8rh_comp_fv_tc.inp	Polynomial hyperelasticity with field variable-dependent coefficients, compressible, hybrid elements, property table.
uhyper_stretch_c3d8rh_incomp_temp_tc.inp	Polynomial hyperelasticity with field variable-dependent coefficients, incompressible, hybrid elements, property table.
uhyper_stretch_cpe4_comp_shear.inp	Neo-Hookean hyperelasticity, compressible, simple shear.
uhyper_stretch_cpe4_comp_tension.inp	Neo-Hookean hyperelasticity, compressible, uniaxial tension.
uhyper_stretch_cps4_comp_temp_tc.inp	Polynomial hyperelasticity with temperature-dependent coefficients, compressible, property table.
uhyper_stretch_cps4r_comp_sdv.inp	Polynomial hyperelasticity, compressible, solution-dependent variables, property table.
uhyper_uhyper_stretch.f	User subroutines *UHYPER* and *UHYPER_STRETCH* used in uhyper_stretch_c3d8_comp_shear.inp, uhyper_stretch_c3d8_comp_tension.inp, uhyper_stretch_cpe4_comp_shear.inp, and uhyper_stretch_cpe4_comp_tension.inp.
uhyper_stretch_tc.f	User subroutine *UHYPER_STRETCH* used in uhyper_stretch_c3d8rh_comp_fv_tc.inp, uhyper_stretch_c3d8rh_incomp_temp_tc.inp, uhyper_stretch_cps4_comp_temp_tc.inp, and uhyper_stretch_cps4r_comp_sdv.inp.

User subroutine for defining different hyperelastic models within the parallel rheological framework

Elements tested

C3D8

Problem description

The tests in this section define different hyperelastic material models in different networks of a nonlinear viscoelastic material model using the parallel rheological framework (see *Parallel Rheological Framework* and **VISCOELASTIC*). For the purpose of verification, the results are compared with those obtained using an element overlay approach.

The element overlay consists of two elements sharing the same nodes. The first element of the overlay has material properties defined by a neo-Hookean hyperelastic material model (see *Table 5*). The second element of the overlay uses a different neo-Hookean hyperelastic model with a nonlinear viscoelastic network, defined such that there is no contribution from the equilibrium network (see *Table 6* and *Table 7*). A third element is introduced to test user subroutine *UHYPER_STRETCH* in combination with a parallel rheological framework definition using the properties described in *Table 5*, *Table 6*, and *Table 7*. The properties and stiffness ratios of the material are defined to provide the exact same response as the overlay model that uses native Abaqus models. The user subroutine defines the strain energy density function with a polynomial hyperelastic model for the equilibrium network (`networkid=0`) and with a neo-Hookean strain energy function for the nonlinear viscoelastic network, (`networkid=1`). All elements are subjected to the same uniaxial tension loading and the same boundary conditions.

Material:

Table 5: Neo-Hookean hyperelastic material 1.

C_{10}	80.0
D_1	0.001

Table 6: Neo-Hookean hyperelastic material 2.

C_{10}	50.0
D_1	0.001

Table 7: Hyperbolic-sine creep model coefficients.

A	2.5×10^{-5}
B	0.3
n	1.0

Results and discussion

The results obtained using user subroutine *UHYPER_STRETCH* are identical to the results obtained using native Abaqus material models and the overlay technique.

Input files

uhyper_stretch_prf_twomaterials.inp	Compressive, uniaxial tension, 3D, parallel rheological framework.
uhyper_stretch_prf_twomaterials.f	User subroutine *UHYPER_STRETCH* used in `uhyper_stretch_prf_twomaterials.inp`.

UINTER

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

CPE4 DC2D4 C3D8T

Features tested

User subroutine to define interfacial constitutive behavior.

Problem description

uinternx.inp

This test verifies that user subroutine *UINTER* properly models hard contact between a deformable and a rigid surface. A single CPE4 element is brought into contact with an analytical rigid surface using displacement boundary conditions. User subroutine *UINTER* models the contact using a penalty approach. The results are compared against those obtained using the default hard contact model in Abaqus/Standard, which uses a Lagrange multiplier-based approach to enforce the contact constraints. It is observed that the penalty approach results in a small penetration of the secondary nodes into the main surface. As a result, there is a difference (about 7.5%) in the contact pressure between the model using *UINTER* and the model using the default hard contact model.

The `lOpenClose` flag is also tested in this problem.

uinternf.inp

This test verifies that user subroutine *UINTER* properly models softened contact along with frictional sliding between a deformable and a rigid surface. The softened contact is modeled using an exponential pressure-clearance relationship, while the shear behavior is modeled using standard Coulomb friction. Both normal and shear behaviors are modeled in user subroutine *UINTER* using a penalty approach. The problem is carried out in two steps. In the first step the deformable body is brought into contact with the rigid surface using boundary conditions. In the second step the deformable body is made to slide on the rigid surface using boundary conditions. The results are compared with a similar problem using the corresponding built-in models in Abaqus/Standard, invoked by specifying the modified contact pressure-overclosure relationship and the friction relationship for the contact surface interaction. The results using the two different approaches (user subroutine *UINTER* versus built-in models) are found to be in good agreement.

uinterht.inp

This test models heat transfer between two surfaces through gap conduction. The model consists of two DC2D4 elements separated by a distance. The two elements are at different initial temperatures. The thermal interaction is modeled using user subroutine *UINTER* by defining the heat flux at the two surfaces as a result of gap conduction. The variations of the heat fluxes with respect to the surface temperatures,

which contribute to the Jacobian, are also specified. The analysis is continued till steady-state conditions are reached. The results are compared with a similar model that uses the built-in capability in Abaqus/Standard to model gap conductance. The results using the two approaches are identical.

uintertd.inp

This test is identical to the verification problem *ufricxxy.inp* (*FRIC*) that uses user subroutine *FRIC* to define the shear interaction between the surfaces, except that both the mechanical and thermal interactions are modeled using user subroutine *UINTER*. It provides verification for using the user subroutine *UINTER* in a fully coupled thermal-stress procedure. As a result of modeling the normal mechanical interaction through *UINTER*, a penalty approach is used in *uintertd.inp*, as opposed to the Lagrange-multiplier-based approach of the built-in hard contact model that is used in *ufricxxy.inp*. The results using the two approaches are in good agreement.

uinterny.inp and uinterim.inp

The test *uinterny.inp* is similar to the test *uinternx.inp*. It includes an extra dummy step in the beginning which is used by *uinterim.inp* to test import of the original model definition. The results of the imported analysis are identical to the results of the original problem. In addition, *uinterny.inp* also includes basic verification for user-defined state variables that are used to store the two local surface directions and the coordinates of the contact point.

Results and discussion

The results in all cases were compared to built-in surface interaction models in Abaqus/Standard and were found to be in good agreement.

Input files

uinternx.inp	Test for surface interaction in the normal direction in a static procedure.
uinternx.f	User subroutine *UINTER* used in uinternx.inp.
uinternf.inp	Test for surface interaction in both the normal and the shear directions in a static procedure.
uinternf.f	User subroutine *UINTER* used in uinternf.inp.
uinterht.inp	Test for thermal surface interaction in a heat transfer procedure.
uinterht_gc.inp	Test for thermal surface interaction in a heat transfer procedure with general contact.
uinterht.f	User subroutine *UINTER* used in uinterht.inp.
uintertd.inp	Test for both mechanical and thermal surface interactions in a coupled temperature-displacement procedure.
uintertd_surf.inp	Test for both mechanical and thermal surface interactions in a coupled temperature-displacement procedure using the surface-to-surface contact formulation.
uintertd.f	User subroutine *UINTER* used in uintertd.inp and uintertd_surf.inp.
uinterny.inp	This test is similar to uinternx.inp. It includes a dummy step in the beginning which is used by uinterim.inp to test import of the original model definition. This problem also includes basic verification for user-defined state variables in user subroutine *UINTER*.
uinterim.inp	Import analysis from uinterny.inp.
uinterny.f	User subroutine *UINTER* used in uinterny.inp and uinterim.inp.

UMAT and UHYPER

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Features tested

User subroutines to define isotropic Mises plasticity, Mooney-Rivlin, and neo-Hookean hyperelastic material properties.

User subroutine for isotropic Mises plasticity model

Elements tested

C3D8

Problem description

This set of verification problems tests many of the variables that are passed into *UMAT*, such as material properties, step times, and strain increment data. These tests also verify that the user-defined quantities of stresses, solution-dependent variables, and the Jacobian matrix are properly transferred to the solution process. These tests are modifications of the tests described in *Rate-independent plasticity*. Wherever an elastic-plastic material was defined in those tests, a user-defined material has been implemented in its place. The structure being analyzed is a cube made of a single C3D8 element.

Material:

Elasticity

Young's modulus, E = 200.0E3

Poisson's ratio, ν = 0.3

Plasticity

Hardening:

Yield stress	Plastic strain
200.	0.0000
220.	0.0009
220.	0.0029

Results and discussion

The tests in this section are set up as cases of homogeneous deformation of a single element of unit dimensions. Consequently, the results are identical for all integration points within the element. In each case the constitutive path is integrated with 20 increments of fixed size.

Input files

umatmst3.inp	Mises plasticity, uniaxial tension, three-dimensional solid.
umatmst3.f	User subroutine *UMAT* used in umatmst3.inp.
umatmss3.inp	Mises plasticity, simple shear, three-dimensional solid.
umatmss3.f	User subroutine *UMAT* used in umatmss3.inp.

User subroutine for hyperelastic model

Elements tested

C3D8 CPE4

Problem description

This set of verification problems is primarily intended to test the deformation gradient that is passed into *UMAT*. Variables in subroutine *UHYPER* that are functions of the deformation gradient are also tested. The structure being analyzed for the two-dimensional case is a unit square made up of three coincident CPE4 elements. The three-dimensional case consists of a cube of unit dimensions made up of three coincident C3D8 elements. For both cases the material properties of the first element are specified directly with general hyperelastic material properties. The same material properties are defined for the second and third elements through user subroutines *UMAT* and *UHYPER*, respectively. The displacements are prescribed at each of the nodes of the models, thus the stresses induced in each element are the same.

Material:

Hyperelasticity: Mooney-Rivlin

C_{10} = 80.0

C_{01} = 0.0

D_1 = 2.013E–4

Results and discussion

The tests in this section are set up as cases of homogeneous deformation of three elements of unit dimensions. Consequently, the results are identical for all integration points within each of the elements. In each case the constitutive path is integrated with 10 increments of fixed size.

Input files

umathrt2.inp	Hyperelasticity, uniaxial tension, two-dimensional solid.
umathrt2.f	User subroutines *UMAT* and *UHYPER* used in umathrt2.inp.
umathrs2.inp	Hyperelasticity, simple shear, two-dimensional solid.
umathrs2.f	User subroutines *UMAT* and *UHYPER* used in umathrs2.inp.
umathrt3.inp	Hyperelasticity, uniaxial tension, three-dimensional solid.
umathrt3.f	User subroutines *UMAT* and *UHYPER* used in umathrt3.inp.
umathrs3.inp	Hyperelasticity, simple shear, three-dimensional solid.
umathrs3.f	User subroutines *UMAT* and *UHYPER* used in umathrs3.inp.

User subroutine for total and incompressible hybrid formulations

Elements tested
C3D8H

Problem description

This set of verification problems is primarily intended to test the total and incompressible hybrid formulations with user subroutine *UMAT*.

Material:

Hyperelasticity: neo-Hookean

C_{10} = 100.0

D_1 = 1.E-03 or D_1 = 0

Results and discussion

The tests in this section are set up as cases of homogeneous deformation. For each case the results are compared against native Abaqus models and are found to be identical to the latter.

Input files

mhncdo3gsh_umat.inp	Almost incompressible hyperelasticity, planar tension, three-dimensional solid.
mhncdo3gsh_umat.f	User subroutine *UMAT* used in mhncdo3gsh_umat.inp.
mhncoo3hut_umat.inp	Fully incompressible hyperelasticity, uniaxial tension, three-dimensional solid.
mhncoo3hut_umat.f	User subroutine *UMAT* used in mhncoo3hut_umat.inp.

User subroutine for frequency domain viscoelastic behavior

Elements tested
C3D8

Problem description

This verification problem is primarily intended to test the implementation of frequency domain viscoelastic behavior in user subroutine *UMAT* in terms of the storage modulus (stiffness) and loss modulus (damping).

Material:

Linear elastic: viscoelastic

E = 200.E+09, ν = 0.3

$(\omega \Re(g^*) = G_\ell / G_\infty)$ = 1.775E-03.

$(\omega \Im(g^*) = 1 - G_s / G_\infty)$ = –5.116E-03

Results and discussion

The test involves the steady-state dynamic response of a cantilever beam subjected to pressure load. The results are compared to a similar model that uses a native Abaqus material model. The results for the two problems are identical.

Input files

visco_freqdmn_umat_ssd.inp	Cantilever beam modeled with C3D8 elements.
visco_freqdmn_umat_ssd.f	User subroutine *UMAT* used in visco_freqdmn_umat_ssd.inp.

Additional user subroutine problems for hyperelastic model

Elements tested

C3D8RH CPS4R

Problem description

This set of verification problems is primarily intended to test the variables that are passed into *UHYPER*. In each test the material properties are specified using general hyperelastic material properties for the first element and the strain energy function and the associated derivatives defined in user subroutine *UHYPER* for the second element. Three different sets of material data are used, as described below.

Material 1: Polynomial (N=1), compressible

C_{10} = 80.0

C_{01} = 20.0

D_1 = 1.E−3

Material 2: Polynomial (N=1), compressible, field variable dependency included for *UHYPER*

C_{10} = 80.0

C_{01} = 20.0

D_1 = 1.5E−3

Material 3: Polynomial (N=1), incompressible, temperature dependency included

θ = 0.0, C_{10} = 80.0, C_{01} = 20.0

θ = 20.0, C_{10} = 75.0, C_{01} = 18.0

θ = 30.0, C_{10} = 70.0, C_{01} = 16.0

Results and discussion

Since the same boundary conditions are applied on the corresponding nodes of the two elements, the results are expected to be identical in the two cases, thus verifying the use of *UHYPER* with the second element.

Input files

uhypercp2s.inp	Compressible, biaxial tension, 2D, state variables, Material 1.
uhypercp2s.f	User subroutine *UHYPER* used in uhypercp2s.inp.
uhyperct3f.inp	Compressible, uniaxial tension, 3D, field variables, Material 2.
uhyperct3f.f	User subroutine *UHYPER* used in uhyperct3f.inp.
uhyperip2t.inp	Incompressible, biaxial tension, 2D, temperature dependency, Material 3.
uhyperip2t.f	User subroutine *UHYPER* used in uhyperip2t.inp.
uhyperip3t.inp	Incompressible, biaxial tension, 3D, temperature dependency, Material 3.
uhyperip3t.f	User subroutine *UHYPER* used in uhyperip3t.inp.

User subroutine for defining different hyperelastic models within the parallel rheological framework

Elements tested

C3D8

Problem description

The tests in this section define different hyperelastic material models in different networks of a nonlinear viscoelastic material model using the parallel rheological framework (see *Parallel Rheological Framework* and **VISCOELASTIC*). For the purpose of verification, the results are compared with those obtained using an element overlay approach.

The element overlay consists of two elements sharing the same nodes. The first element of the overlay has material properties defined by a polynomial hyperelastic material model (see *Table 1*). The second element of the overlay uses a neo-Hookean hyperelastic model with a nonlinear viscoelastic network, defined such that there is no contribution from the equilibrium network (see *Table 2* and *Table 3*). A third element is introduced to test user subroutine *UHYPER* in combination with a parallel rheological framework definition using the properties described in *Table 1*, *Table 2*, and *Table 3*. The properties and stiffness ratios of the material are defined to provide the exact same response as the overlay model that uses native Abaqus models. The user subroutine defines the strain energy density function with a polynomial hyperelastic model for the equilibrium network (`networkid=0`) and with a neo-Hookean strain energy function for the nonlinear viscoelastic network, (`networkid=1`). All elements are subjected to the same uniaxial tension loading and the same boundary conditions.

Material:

Table 1: Polynomial hyperelastic material (N=1).

C_{10}	80.0
C_{01}	20.0
D_1	0.001

Table 2: Neo-Hookean hyperelastic material.

C_{10}	50.0

D_1	0.001

Table 3: Hyperbolic-sine creep model coefficients.

A	2.5×10^{-5}
B	0.3
n	1.0

Results and discussion

The results obtained using user subroutine *UHYPER* are identical to the results obtained using native Abaqus material models and the overlay technique.

Input files

uhyper_tension_prf_twomaterials.inp	Compressive, uniaxial tension, 3D, parallel rheological framework.
uhyper_tension_prf_twomaterials.f	User subroutine *UHYPER* used in `uhyper_tension_prf_twomaterials.inp`.

UMATHT

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

DC2D4

Features tested

User subroutine to define a simple thermal constitutive behavior.

Problem description

User subroutine *UMATHT* is used to define the thermal behavior of the material for a transient heat transfer analysis. Isotropic conductivity and a constant specific heat for the material are assumed. The heat conduction in the material is governed by Fourier's law, and the gradient of the heat flux with respect to temperature is zero. The material properties—namely, conductivity and specific heat—are defined and are passed into the user subroutine via the `PROPS` array.

Verification problem *ec24dfd2.inp* solves the same problem using the standard data line options to define conductivity and specific heat.

Results and discussion

The results of *umathtxx.inp* match those obtained from *ec24dfd2.inp*.

Input files

umathtxx.inp	Input file for this analysis.
umathtxx.f	User subroutine *UMATHT* used in umathtxx.inp.

URDFIL

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

B21 C3D20P CPE4R CPE4RH CPE4T DC2D4 S3R S8R5

Features tested

User subroutine to allow reading of the results file at the end of any increment in an analysis.

Problem description

This set of verification problems ensures that the subroutine *URDFIL* is called properly for the various procedure types. In each problem the utility routine `POSFIL` is called from within *URDFIL*. Use is made of the `LSTOP` and `LOVRWRT` parameters such that the results file differs from that which would be produced by the same analysis without *URDFIL*.

Results and discussion

The results file can be checked to ensure that *URDFIL* and `POSFIL` are functioning correctly for the various procedures.

Input files

ufilcpl.inp	Coupled temperature-displacement analysis.
ufilcpl.f	User subroutine *URDFIL* used in ufilcpl.inp.
ufildyn.inp	Dynamic analysis.
ufildyn.f	User subroutine *URDFIL* used in ufildyn.inp.
ufilhtr.inp	Heat transfer analysis.
ufilhtr.f	User subroutine *URDFIL* used in ufilhtr.inp.
ufilmod.inp	Modal dynamic analysis.
ufilmod.f	User subroutine *URDFIL* used in ufilmod.inp.
ufilrre.inp	Random response analysis.
ufilrre.f	User subroutine *URDFIL* used in ufilrre.inp.
ufilrsp.inp	Response spectrum analysis.
ufilrsp.f	User subroutine *URDFIL* used in ufilrsp.inp.
ufilsoi.inp	Soils consolidation analysis.
ufilsoi.f	User subroutine *URDFIL* used in ufilsoi.inp.
ufilssd.inp	Steady-state dynamics analysis.
ufilssd.f	User subroutine *URDFIL* used in ufilssd.inp.

ufilsta.inp	Statics analysis.
ufilsta.f	User subroutine *URDFIL* used in ufilsta.inp.
ufilvis.inp	Visco analysis.
ufilvis.f	User subroutine *URDFIL* used in ufilvis.inp.

USDFLD

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

B21 CPE4 DC1D2 DC2D4 DS4 S4 S4R

Features tested

User subroutine to redefine field variables at a material point.

Problem description

This set of tests verifies that field variable values are properly transferred to a structure when the values are redefined at run time. In every instance an Abaqus material model with dependency on a solution variable (such as temperature or equivalent plastic strain) is implemented with field-variable dependence. The appropriate field values are computed at run-time based on solution values from the previous increment. Every user-defined field variable model is checked against the equivalent Abaqus material model.

The hypoelastic material model is chosen as the basis for nonlinear elastic behavior at small strains for both static and dynamic analyses. Since Abaqus does not provide dependence of the hypoelastic tangent modulus on field variables, it was implemented using an elastic material with the equivalent secant modulus.

Results and discussion

A very close match is obtained between the user field variable approach and the corresponding Abaqus model. *Figure 1* shows how the hypoelastic models compare in a static analysis. Matches of a similar nature can be obtained for the other files by using the time-history plotting capability available in the Visualization module of Abaqus/CAE.

Since the field variable approach uses values from the previous increment, the solution should improve as the time increment decreases. This trend was observed throughout.

Input files

udfcd1hs.inp	Steady-state heat transfer analysis, DC1D2 elements.
udfcd1hs.f	User subroutine *USDFLD* used in udfcd1hs.inp.
udfcd1ht.inp	Transient heat transfer analysis, DC1D2 elements.
udfcd1ht.f	User subroutine *USDFLD* used in udfcd1ht.inp.
udfcd2hs.inp	Steady-state heat transfer analysis, DC2D4 elements.
udfcd2hs.f	User subroutine *USDFLD* used in udfcd2hs.inp.
udfcd2ht.inp	Transient heat transfer analysis, DC2D4 elements.
udfcd2ht.f	User subroutine *USDFLD* used in udfcd2ht.inp.
udfcdshs.inp	Steady-state heat transfer analysis, DS4 elements.

udfcdshs.f	User subroutine *USDFLD* used in udfcdshs.inp.
udfcdsht.inp	Transient heat transfer analysis, DS4 elements.
udfcdsht.f	User subroutine *USDFLD* used in udfcdsht.inp.
udfebxdx.inp	Dynamic analysis, elastic, B21 elements.
udfebxdx.f	User subroutine *USDFLD* used in udfebxdx.inp.
udfebxsx.inp	Static analysis, elastic, B21 elements.
udfebxsx.f	User subroutine *USDFLD* used in udfebxsx.inp.
udfecxdx.inp	Dynamic analysis, elastic, CPE4 elements.
udfecxdx.f	User subroutine *USDFLD* used in udfecxdx.inp.
udfecxsx.inp	Static analysis, elastic, CPE4 elements.
udfecxsx.f	User subroutine *USDFLD* used in udfecxsx.inp.
udfesxdx.inp	Dynamic analysis, elastic, S4R elements.
udfesxdx.f	User subroutine *USDFLD* used in udfesxdx.inp.
udfesxsx.inp	Static analysis, elastic, S4/S4R elements.
udfesxsx.f	User subroutine *USDFLD* used in udfesxsx.inp.
udfpbxsf.inp	Static analysis followed by frequency analysis, elastic-plastic, B21 elements.
udfpbxsf.f	User subroutine *USDFLD* used in udfpbxsf.inp .
udfpcxsf.inp	Static analysis followed by frequency analysis, elastic-plastic, CPE4 elements.
udfpcxsf.f	User subroutine *USDFLD* used in udfpcxsf.inp.
udfpsxsf.inp	Static analysis followed by frequency analysis, elastic-plastic, S4/S4R elements.
udfpsxsf.f	User subroutine *USDFLD* used in udfpsxsf.inp.

Figures

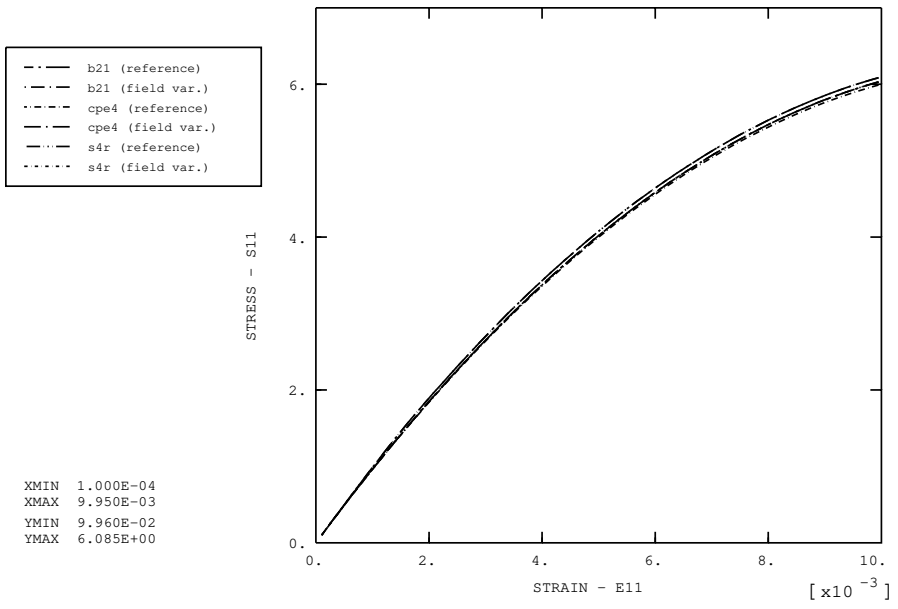


Figure 1: Hypoelastic model comparison.

UTEMP, VUTEMP, UFIELD, UMASFL, and UPRESS

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Features tested

User subroutines to define temperatures, field variables, mass flow rates, and equivalent pressure stresses.

Setting temperature and field data using user subroutines

Elements tested

T3D2

Problem description

This set of tests verifies that temperature and field variable values are properly transferred to a structure when the values are set using user subroutines. These tests are modifications of the tests described in *Defining temperature, field variable, pressure stress, and pore fluid pressure values*. Wherever results files were used in those tests, most have been replaced here with user subroutines. The structure being analyzed is a cantilevered truss made up of 10 T3D2 elements.

The tests are as follows:

utmpfvs1.inp

This file tests setting temperature and more than one field variable using user subroutines. The variation of temperature and all three field variables are linear with time as follows:

	Initial value	Final value
Temperature	100	200
Field variable 1	100	200
Field variable 2	200	250
Field variable 3	100	200

utmpfvs2.inp

This file tests setting a field variable from a user subroutine without temperature being present in the problem. This is an important test because of the way that temperatures and field variables are stored internally. The field variable varies linearly with time, as follows:

	Initial value	Final value
Field variable	100	200

(The problem that is analogous to test *xtfvtrs3.inp* in *Defining temperature, field variable, pressure stress, and pore fluid pressure values* is omitted, since this analysis would not test any features that were not already covered by the other tests in this section.)

utmpfvs4.inp

This is a three-step problem involving temperature and one field variable. In the first step an amplitude curve is used to set temperature to 200 and the field variable to 250. In the second step temperature and the field variable are set twice: first, values are read from results files, and then the user subroutines multiply all values by two. This results in ramping the temperature to 400 and the field variable to 500 over the step. The results files used are as follows:

xtfvtrt1.fil ⇒ Temperature

xtfvtrt2.fil ⇒ Field variable 1

(These two heat transfer problems are described further in *Defining temperature, field variable, pressure stress, and pore fluid pressure values*.) In the third step both the temperature and the field variable are reset to their initial conditions.

The following must be confirmed by this test:

- The user subroutine must mesh smoothly with other methods of setting temperature and field variables used in other steps.
- The user subroutine must have access to values set from a results file and must be able to modify those values.
- If temperature or a field variable is set by data line input and then modified by a user subroutine within the same step, the values given on the data lines must be ignored.
- The variable KSTEP must be available for use in both user subroutines.

utmpfvsr.inp

This analysis restarts *utmpfvs4.inp* from the third step. Temperature and the field variable are both set using user subroutines as follows:

	Initial value	Final value
Temperature	0	100
Field variable	0	50

utmpfvsn.inp

This file tests setting all of the field variables simultaneously in user subroutine *UFIELD*. The final results are the same as those obtained in *utmpfvs1.inp*.

Results and discussion

The only quantities of interest are the temperatures and field variables in the structure. Expected solutions are shown in *Figure 1* through *Figure 3*.

Input files

utmpfvs1.inp	Stress analysis, first run.
utmpfvs1.f	User subroutines *UTEMP* and *UFIELD* used in utmpfvs1.inp.
utmpfvs2.inp	Stress analysis, second run.
utmpfvs2.f	User subroutine *UFIELD* used in utmpfvs2.inp.
utmpfvs4.inp	Stress analysis, analogous to xtfvtrs4.inp.

utmpfvs4.f	User subroutines *UTEMP* and *UFIELD* used in utmpfvs4.inp.
utmpfvsr.inp	Stress analysis, restart of utmpfvs4.inp.
utmpfvsr.f	User subroutines *UTEMP* and *UFIELD* used in utmpfvsr.inp.
utmpfvsn.inp	Stress analysis, NUMBER.
utmpfvsn.f	User subroutines *UTEMP* and *UFIELD* used in utmpfvsn.inp.

Figures

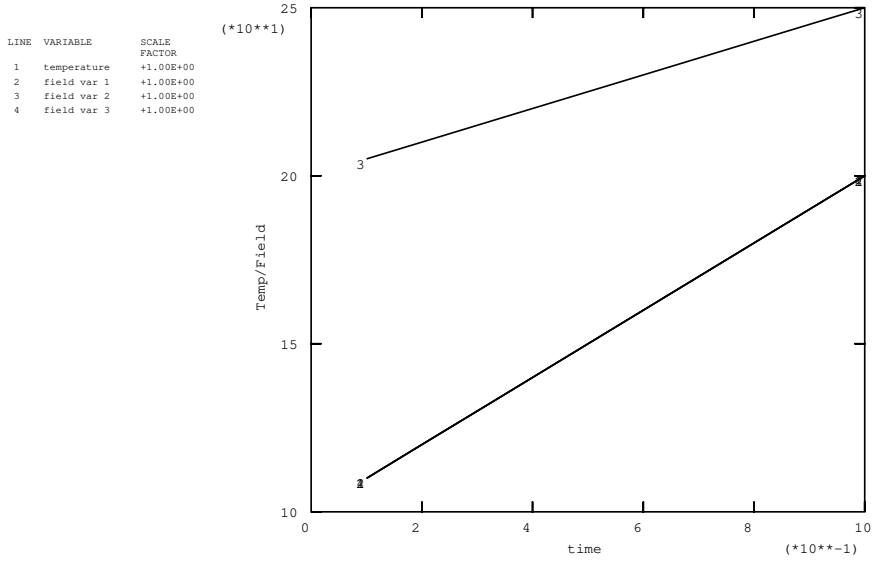


Figure 1: Temperature and field variables for `utmpfvs1.inp`.

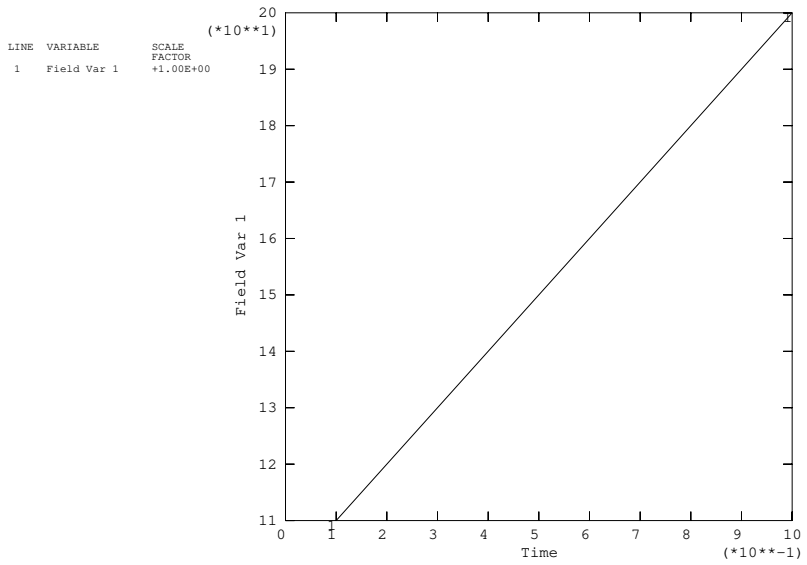


Figure 2: Field variable for `utmpfvs2.inp`.

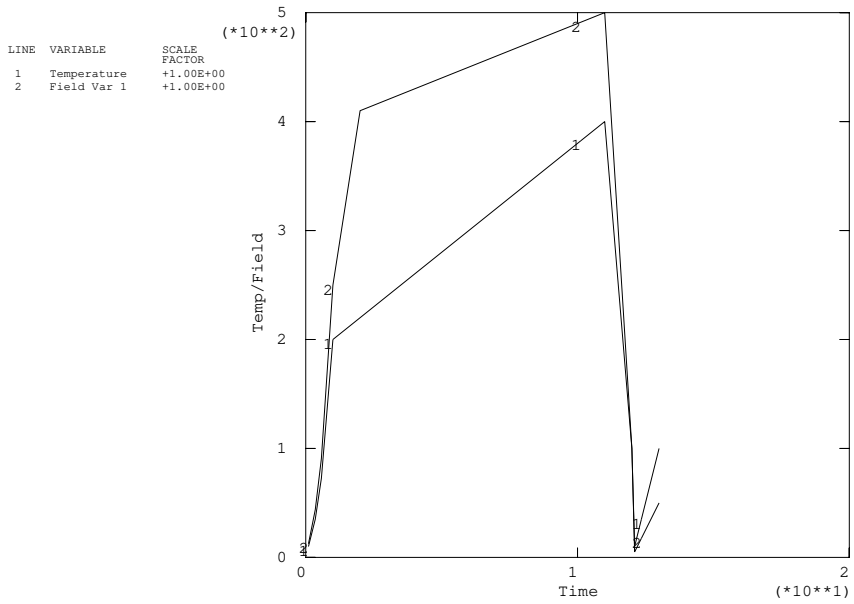


Figure 3: Temperatures and field variable for `utmpfvs4.inp` and `utmpfvsr.inp`.

Composite shell temperature loads from user subroutines

Elements tested

S4R S4R5 S8R5

Problem description

This set of tests verifies the use of user subroutines *UTEMP*, *VUTEMP*, and *UFIELD* in conjunction with composite structural shells. These tests are modifications of the tests described in *Defining temperature, field variable, pressure stress, and pore fluid pressure values*. Values that were obtained from results files in those problems are set here with user subroutines. A three-layered composite shell with a prescribed temperature or field variable profile through the cross-section is analyzed. Three temperature points and five section integration points are used for each layer. The temperature and field variables are assigned to these five points through a linear interpolation of the three values available per layer from the user subroutine. The results of these analyses verify that this interpolation is correct.

The user subroutines are tested for 4-node shells and 8-node shells.

Results and discussion

The temperature and field variable profiles were chosen to be identical to those obtained in heat transfer problems *xtmpcst4.inp* and *xtmpcst8.inp*, so that the results of the stress analyses could be directly compared with results from *xtmpcss4.inp*, *xtmpcss8.inp*, *xfvcss4x.inp*, and *xfvcss8x.inp*. (For a description of the heat transfer problem, see *Defining temperature, field variable, pressure stress, and pore fluid pressure values*.) The temperature/field variable profile is as follows:

- The temperature/field variable at the bottom of layer 1 is 425.0°.
- The temperature/field variable at the top of layer 1 and the bottom of layer 2 is 373.2°.
- The temperature/field variable at the top of layer 2 and the bottom of layer 3 is 336.8°.
- The temperature/field variable at the top of layer 3 is 287.5°.

There is a linear variation between the top and bottom of each layer.

It can be seen that the temperature and field variable values are properly transferred to the structural composite shell.

Input files

utempc4x.inp	*UTEMP*, S4R5 elements.
utempc4x.f	User subroutine *UTEMP* used in utempc4x.inp.
UserTemp_CompShell_xpl.inp	User subroutine *VUTEMP*, S4R elements.
Vutemp_CompShell_xpl.f	User subroutine *VUTEMP* used in `UserTemp_CompShell_xpl.inp`.
ufieldc4.inp	*UFIELD*, S4R5 elements.
ufieldc4.f	User subroutine *UFIELD* used in ufieldc4.inp.
utempc8x.inp	*UTEMP*, S8R5 elements.
utempc8x.inp	User subroutine *UTEMP* used in utempc8x.inp.
ufieldc8.inp	*UFIELD*, S8R5 elements.
ufieldc8.f	User subroutine *UFIELD* used in ufieldc8.inp.

Gap conductance problems with field variables and mass flow rates set using user subroutines UFIELD and UMASFL

Elements tested

C3D8T DC3D8 DCC3D8 DINTER4 INTER4T

Problem description

These tests verify that field variables and mass flow rates are properly transferred to a structure during heat transfer and coupled temperature-displacement analyses. These tests are modifications of the tests described in *Thermal properties* and *GAPCON*. The tests are cases of uniform one-dimensional heat flux using three-dimensional elements. Consequently, the temperature results are identical for all nodes located at a particular plane along the direction of heat flow. In all cases a steady-state heat transfer analysis is performed in several increments. Values of predefined field variables or mass flow rates vary during the solution, which in turn influences the conductivity across the interface and, thus, the solution.

Results and discussion

The results match the exact solutions.

Input files

ufieldghs.inp	Field-variable-dependent gap conductivity, heat transfer analysis, DC3D8 and DINTER4 elements.
ufieldghs.f	User subroutine *UFIELD* used in ufieldghs.inp.
umasflghs.inp	Mass-flow-rate-dependent gap conductivity, heat transfer analysis, DCC3D8 and DINTER4 elements.
umasflghs.f	User subroutine *UMASFL* used in umasflghs.inp.
ufieldgcs.inp	Field-variable-dependent gap conductivity, coupled temperature-displacement analysis, C3D8T and INTER4T elements.
ufieldgcs.f	User subroutine *UFIELD* used in ufieldgcs.inp.

Mass diffusion problems with pressure stresses set using user subroutine UPRESS

Elements tested

DC3D8 DC3D20

Problem description

These tests verify that equivalent pressure stresses are transferred properly to a structure during a mass diffusion analysis. The tests are cases of uniform one-dimensional mass diffusion using three-dimensional elements. Consequently, the concentration results are identical for all nodes located at a particular plane along the diffusion direction.

Results and discussion

The results match the exact solutions.

Input files

upress38.inp	DC3D8 elements.
upress38.f	User subroutine *UPRESS* used in upress38.inp.
upress20.inp	DC3D20 elements.
upress20.f	User subroutine *UPRESS* used in upress20.inp.

UTRSNETWORK

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D8

Features tested

User subroutine to define a reduced time shift function for models defined within the parallel rheological framework.

Problem description

The input file consists of one element with the material response specified using the parallel rheological framework with two viscous networks. The reduced times were used with the shift function approximated by the Williams-Landel-Ferry form in network one and by the Arrhenius form in network two, both defined in user subroutine *UTRSNETWORK*. The element is preloaded in the first static step in which viscous effects are not active. In the second quasi-static step the viscous effects are activated and the load is held constant so that the material is subjected to creep. The temperature is also increased during the second step. The results were compared to the results obtained with an almost identical model that uses the same shift functions specified using the built-in Abaqus functionality.

Results and discussion

The results obtained using the user subroutine and the built-in Abaqus models are identical.

Input files

viscnet_c3d8_nh_n2_trs_user.inp	The model that uses user subroutine *UTRSNETWORK* to define the Williams-Landel-Ferry and Arrhenius shift functions.
viscnet_wlf_arrhenius_trs.f	User subroutine to define the Williams-Landel-Ferry and Arrhenius forms of the shift function.

UVAREL

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D8 C3D8R CPE4
S4 S4R

Features tested

User subroutine to define element user output variables as functions of standard output variables.

Problem description

This set of verification problems tests many of the variables that are passed into *UVAREL*, as well as integration point quantities that are read by utility subroutine `getElMPtVar`. Most of the tests are single-element cases that set user-defined element output variables directly equal to output quantities such as stresses and section forces. These tests also verify that the user-defined output variables are transferred properly to the solution process.

Results and discussion

The results verify that user subroutine *UVAREL* picks up the correct variables and transfers them properly to the output files.

Input files

uvare_cpe4.inp	CPE4 element; S.
one_uvarel.f	User subroutine *UVAREL* used in `uvare_cpe4.inp`, `uvare_cpe4_label.inp`, `uvare_s4.inp`, `uvare_compshell.inp`, and `uvare_compsolid.inp`.
uvare_cpe4_label.inp	CPE4 element; labels and descriptions; S.
uvare_s4.inp	S4 element; S.
uvare_compshell.inp	S4 element; composite layers, S.
uvare_compsolid.inp	C3D8 element; S.
uvare_compshell_sf.inp	S4 element; utility subroutines `getUVarElOrientationName` and `getUVarElMaterialName`; SF, STH, SSAVG.
one_uvarel_sf.f	User subroutine *UVAREL* used in `uvare_compshell_sf.inp`.
uvare_sdv.inp	C3D8R element; SDV, UVARM.
one_uvarel_sdv.f	User subroutine *UVAREL* used in `uvare_sdv.inp`.
one_uvarel_sdv.cpp	User subroutine *UVAREL* used in `uvare_sdv.inp` to verify C/C++ interfaces to user utilities.
uvare_compshell_mfh.inp	S4R element; composite layers; mean-field homogenization; S.

one_uvarel_mfh.f	User subroutine *UVAREL* used in `uvare_compshell_mfh.inp`.

UVARM

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

B21 B31
C3D8 C3D8R C3D8T CAX4 CAX4E CAXA42 CPE4R CPE4T CPE8RP CPS4R
DC1D2 DC2D4 DC3D8
M3D4 M3D4R
S4 S4R S8R SAX1 SAXA12
T3D2 T3D3T

Features tested

User subroutine to define user output variables as functions of standard output variables.

Problem description

This set of verification problems tests many of the variables that are passed into *UVARM*, as well as integration point quantities that are read by the utility subroutine GETVRM. Most of the tests are single-element cases that set user-defined output variables directly equal to integration point quantities such as stresses and strains. These tests also verify that the user-defined output variables are transferred properly to the solution process.

Results and discussion

The results verify that the subroutine GETVRM picks up the correct variables and transfers them properly to the output files.

Input files

Heat transfer analyses:

uvarhfv3.inp	DC3D8 elements; field-variable-dependent conductivity; HFL, TEMP, FV, GRADT.
uvarhfv3.f	User subroutine *UVARM* used in uvarhfv3.inp.
uvarhfv2.inp	DC2D4 elements; field-variable-dependent conductivity; HFL, TEMP, FV, GRADT.
uvarhfv2.f	User subroutine *UVARM* used in uvarhfv2.inp.
uvarhfv1.inp	DC1D2 elements; field-variable-dependent conductivity; HFL, TEMP, FV, GRADT.
uvarhfv1.f	User subroutine *UVARM* used in uvarhfv1.inp.

Coupled temperature-displacement analyses:

uvartel3.inp	C3D8T elements; linear elastic; S, HFL, TEMP, COORD, THE, THEP, GRADT.

uvartel3.f	User subroutine *UVARM* used in uvartel3.inp.
uvartel2.inp	CPE4T elements; linear elastic; S, HFL, TEMP, COORD, THE, THEP, GRADT.
uvartel2.f	User subroutine *UVARM* used in uvartel2.inp.
uvartel1.inp	T3D3T elements; linear elastic; S, HFL, TEMP, COORD, THE, THEP, GRADT.
uvartel1.f	User subroutine *UVARM* used in uvartel1.inp.
Geostatic analyses:	
uvargdp3.inp	CAX4 elements; Drucker-Prager; E, EP, PE, PEP, NE, NEP, LE, LEP.
uvargdp3.f	User subroutine *UVARM* used in uvargdp3.inp.
uvargdp2.inp	CPE4R elements; Drucker-Prager; E, EP, PE, PEP, NE, NEP, LE, LEP.
uvargdp2.f	User subroutine *UVARM* used in uvargdp2.inp.
Static analyses:	
uvarsel3.inp	B31 elements; linear elastic; S, E, IVOL, TEMP.
uvarsel3.f	User subroutine *UVARM* used in uvarsel3.inp.
uvarsel2.inp	B21 elements; linear elastic; S, E, IVOL, TEMP.
uvarsel2.f	User subroutine *UVARM* used in uvarsel2.inp.
uvarself.inp	S4R elements; linear elastic; S, STH, CFAILURE, ENER.
uvarself.f	User subroutine *UVARM* used in uvarself.inp.
uvarsele.inp	S4 elements; linear elastic; S, STH, CFAILURE, ENER.
uvarsele.f	User subroutine *UVARM* used in uvarsele.inp.
uvarseln.inp	CAXA42 elements; linear elastic; S, SINV, E.
uvarseln.f	User subroutine *UVARM* used in uvarseln.inp.
uvarsecn.inp	SAXA12 elements; elastic, composite; S, SP, SINV, E, EP, ENER.
uvarsecn.f	User subroutine *UVARM* used in uvarsecn.inp.
uvarseca.inp	SAX1 elements; elastic, composite; S, SP, CFAILURE, ENER.
uvarseca.f	User subroutine *UVARM* used in uvarseca.inp.
uvarsec2.inp	S8R elements; elastic, composite; S, SP, TSHR, CFAILURE, ENER.
uvarsec2.f	User subroutine *UVARM* used in uvarsec2.inp.
uvarsmp3.inp	C3D8 elements; Mises plasticity; ALPHA, ALPHAP, SINV, E, EP, PE, PEP, ENER.
uvarsmp3.f	User subroutine *UVARM* used in uvarsmp3.inp.
uvarsmp2.inp	CPS4R elements; Mises plasticity; ALPHA, ALPHAP, SINV, E, EP, PE, PEP, ENER.
uvarsmp2.f	User subroutine *UVARM* used in uvarsmp2.inp.
uvarsmp1.inp	T3D2 elements; Mises plasticity; ALPHA, ALPHAP, SINV, E, EP, PE, PEP, ENER.
uvarsmp1.f	User subroutine *UVARM* used in uvarsmp1.inp.
uvarscn3.inp	C3D8R elements; concrete; S, SP, PE, PEP, DG, DGP, CRACK.
uvarscn3.f	User subroutine *UVARM* used in uvarscn3.inp.
uvarscn2.inp	CPS4R elements; concrete; S, SP, PE, PEP, DG, DGP, CRACK.
uvarscn2.f	User subroutine *UVARM* used in uvarscn2.inp.
uvarscn1.inp	T3D2 elements; concrete; S, SP, DG, DGP, CRACK.

uvarscn1.f	User subroutine *UVARM* used in uvarscn1.inp.
uvarsep2.inp	CPE8RP elements; porous elastic; SAT, POR, GELVR, VOIDR.
uvarsep2.f	User subroutine *UVARM* used in uvarsep2.inp.
uvarsep3.inp	CPE8RP elements; porous elastic with anisotropic swelling; SAT, POR, GELVR, VOIDR.
uvarsep3.f	User subroutine *UVARM* used in uvarsep3.inp.
uvarspz3.inp	CAX4E elements; piezoelectric; S, E, EFLX, EPG.
uvarspz3.f	User subroutine *UVARM* used in uvarspz3.inp.
uvarscp3.inp	C3D8R elements; Cap plasticity; S, SP, IE, IEP, EE, EEP, PEQC.
uvarscp3.f	User subroutine *UVARM* used in uvarscp3.inp.
uvarsreb.inp	C3D8R, S4R, M3D4R elements; elastic, rebar; S, E, IVOL, RBFOR, RBANG, RBROT.
uvarsreb.f	User subroutine *UVARM* used in uvarsreb.inp.

Static perturbation analyses:

uvar_se_3d_pert.inp	C3D8R; linear elastic; E, ENER, EP, S, SINV, SP.
uvar_se_3d_pert.f	User subroutine *UVARM* used in uvar_se_3d_pert.inp.
uvar_se_beam_pert.inp	B31; linear elastic; E, ENER, EP, S, SINV, SP.
uvar_se_beam_pert.f	User subroutine *UVARM* used in uvar_se_beam_pert.inp.
uvar_se_plane_strain_pert.inp	CPE4R; linear elastic; E, ENER, EP, S, SINV, SP.
uvar_se_plane_strain_pert.f	User subroutine *UVARM* used in uvar_se_plane_strain_pert.inp.
uvar_se_plane_stress_pert.inp	CPS4R; linear elastic; E, ENER, EP, S, SINV, SP.
uvar_se_plane_stress_pert.f	User subroutine *UVARM* used in uvar_se_plane_stress_pert.inp.
uvar_temp_3d_pert.inp	C3D8R; linear elastic; TEMP, THE, THEP.
uvar_temp_3d_pert.f	User subroutine *UVARM* used in uvar_temp_3d_pert.inp.
uvar_pz_axi_pert.inp	CAX4E; linear elastic; EFLX, ENER, EPG.
uvar_pz_axi_pert.f	User subroutine *UVARM* used in uvar_pz_axi_pert.inp.

Dynamic analyses:

uvardel3.inp	C3D8R elements; linear elastic; S, SINV, ER, ERP, ENER, COORD.
uvardel3.f	User subroutine *UVARM* used in uvardel3.inp.
uvardel2.inp	CPS4R elements; linear elastic; S, SINV, ER, ERP, ENER, COORD.
uvardel2.f	User subroutine *UVARM* used in uvardel2.inp.
uvardel1.inp	T3D2 elements; linear elastic; S, SINV, ER, ERP, ENER, COORD.
uvardel1.f	User subroutine *UVARM* used in uvardel1.inp.

Visco analyses:

uvarvve3.inp	C3D8R elements; viscoelastic; S, SP, E, EP, CE, CEP, ENER.
uvarvve3.f	User subroutine *UVARM* used in uvarvve3.inp.
uvarvve2.inp	CPS4R elements; viscoelastic; S, SP, E, EP, CE, CEP, ENER.
uvarvve2.f	User subroutine *UVARM* used in uvarvve2.inp.

UWAVE and UEXTERNALDB

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Features tested

User subroutine *UWAVE* is used to specify wave kinematics for an Abaqus/Aqua analysis, and user subroutine *UEXTERNALDB* is used to manipulate external user files in the same analysis.

uwavexx1.inp

Elements tested

B31 PIPE31

Problem description

Verification problem *uwavexx1.inp* is identical to *ep32pxx1.inp* as described in *Dynamic pressure, closed-end buoyancy loads* of *Aqua load cases*. It tests the dynamic pressure implementation and closed-end buoyancy loading for Airy waves coded in user subroutine *UWAVE*. In this problem a vertical pile is fully constrained and subjected to buoyancy loading. The Airy wave theory is used to calculate the total reaction force on the structure during a dynamic stress/displacement analysis by direct integration. Distributed load type PB is used with a 50-element (PIPE31) model, and concentrated load type TSB is used with a one-element (B31) model.

Model:

Height of the structure	175.0 (100.0 below and 75.0 above mean water elevation)
Pipe section data	$r = 1.0$, $t = 0.25$

Material:

Young's modulus	1×10^6

Aqua – environment:

Seabed elevation	100.0
Mean water elevation	1100.0
Gravitational constant	32.2
Fluid mass density	2.0

Results and discussion

The results are identical to those calculated in *ep32pxx1.inp* and agree well with the analytically determined peak total reaction force.

Input files

uwavexx1.inp	Input file for this analysis.
uwavexx1.f	User subroutine *UWAVE* used in uwavexx1.inp.

uwavexx2.inp

Elements tested

B21

Problem description

The input file for verification problem *uwavexx2.inp* is identical to file *riserdynamics_airy_disp.inp* used in *Riser dynamics*, except for the additional specification of gravity waves using user subroutine *UWAVE* and the output request for the nodal forces. The purpose of adding these items will be made clear in the problem description for *uwavexx3.inp* that follows. In this example a riser is modeled with 10 beam elements of type B21. The riser is subjected to self-weight, a top tension load, and drag loading caused by a steady current flowing by it. Waves with a peak-to-trough height of 6.1 m (20 ft) travel across the water surface with a period of 9 seconds; these waves are modeled with the Airy wave theory coded in user subroutine *UWAVE*.

Results and discussion

The results are identical to those calculated in the analysis using the built-in Airy wave option (file *riserdynamics_airy_disp.inp*), discussed in *Riser dynamics*.

Input files

uwavexx2.inp	Input file for this analysis.
uwavexx2.f	User subroutines *UWAVE* and *DISP* used in uwavexx2.inp.

uwavexx3.inp

Elements tested

B21

Problem description

This is a multipurpose verification problem for stochastic wave analysis with user subroutine *UWAVE*. The first of the four objectives is to test the restart capability; accordingly, the job is set up to rerun the dynamic analysis (Step 2) in *uwavexx2.inp*. The second objective is to exercise the coding for stochastic wave analysis, which is invoked with gravity waves using user subroutine *UWAVE*. A random number seed and the wave spectrum (wave frequency/amplitude data pairs) can be specified. These data are optional; dummy values for the random number seed and the wave spectrum are specified here to verify that they are accessed correctly in subroutine *UWAVE* during the analysis.

For stochastic wave analysis an intermediate configuration is available to *UWAVE*. This intermediate configuration can be used to store a user-chosen configuration upon which the wave kinematics are based. The third objective of this problem is to exercise the coding that performs the updating of the intermediate configuration. To this end, the *UWAVE* routine specifies that for the dynamic analysis (Step 2) a global update be performed for all elements at increments 1 and 141; and a local update be performed for a single element at 10-increment intervals, starting at increment 11 for element 1 and finishing at increment 101 for element 10. The local updates are requested sequentially for elements 1 through 10. For each global and local update request, the code updates the

intermediate configuration with the current configuration for all the elements in the model and for the individual element, respectively. In this second step the updated intermediate configuration is stored in a common block array for manipulation in Step 3.

Step 3 is a dummy step used to overwrite the NFORC records with intermediate configuration information. When *UWAVE* is executed for this step, the intermediate configuration data stored in the common block are saved to an external file (`UWAVEXX3.017`) for subsequent postprocessing. To facilitate internal QA of the intermediate configuration data, the program `UWAVEXX5.f` is run to transfer the data from `UWAVEXX3.017` to `UWAVEXX3.fil` by overwriting the dummy NFORC records created in Step 2. The resulting file, `UWAVEXX3.fin`, is then renamed to `UWAVEXX4.fil` to allow for data manipulation via accessing the NFORC records.

The last objective is to test the *UEXTERNALDB* user subroutine interface. This subroutine can be used to manage user-defined external databases. When this subroutine is called at the beginning of the analysis, it allows for opening external user files and initialization of external user common blocks. When this subroutine is called at the end of the analysis, it allows for closing open external files. In this example the *UEXTERNALDB* subroutine creates, opens, and writes to the file `UWAVEXX3.96` using Fortran unit 96. The dummy wave spectrum data are written to this file.

Results and discussion

The dynamic analysis (Step 2) results obtained by the restart analysis are identical to those obtained in *uwavexx2.inp*. The dummy NFORC records written to the `UWAVEXX3.fin` (or `UWAVEXX4.fil`) file are verified to contain the requested intermediate configurations. External user file manipulation in user subroutine *UEXTERNALDB* is also exercised.

Input files

uwavexx3.inp	Input file for this analysis.
uwavexx3.f	User subroutines *UWAVE*, *DISP*, and *UEXTERNALDB* used in uwavexx3.inp.

VDISP

Products: Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

T3D2

Features tested

User subroutine to provide prescribed nodal behavior (displacements, velocities, and accelerations).

Problem description

A straight section built with one-dimensional truss elements is used in a dynamic analysis. The model has a displacement boundary condition prescribed at node 2, a velocity boundary condition prescribed at node 3, and an acceleration boundary condition prescribed at node 4 using user subroutine *VDISP*. For comparison purposes a displacement variation is specified at node 5, a velocity variation is specified at node 6, and an acceleration variation is specified at node 7 using amplitude functions. The variation prescribed is

$$f(t) = 1.5 \quad \sin(.1\pi t)$$

for displacement and

$$f(t) = 1.5\sin(.1\pi t) + .5\cos(.1\pi t)$$

for velocity and acceleration. The cosine contribution is excluded in selecting the displacement amplitude function to avoid an initial jump in the displacement. For the variations specified using *VDISP*, the appropriate functions have to be incorporated into the subroutine. Identical variations are specified in both methods such that the results should be identical.

Results and discussion

The responses of the nodal degrees of freedom can be plotted to show that user subroutine *VDISP* is providing the same history as the amplitude function.

Input files

vdisp_uva.inp	Input file for this analysis.
vdisp_uva.f	User subroutine *VDISP* used in vdisp_uva.inp.

VDLOAD

Products: Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Nonuniform body forces

Elements tested

CPE3 CPE4R CPS3 CPS4R CAX3 CAX4R C3D4 C3D6 C3D8R
M3D3 M3D4R SAX1 S3R S4R T2D2 T3D2

Features tested

Nonuniform body forces.

Problem description

In this verification test all the available element types are tested by loading them with a nonuniform body force. All the element nodes are fixed in position, and the reaction forces generated at the nodes are used to verify the element load calculations. The purpose of this test is to verify the element load calculations, not to test all the capabilities of user subroutine *VDLOAD*.

The material model is isotropic linear elasticity. The material properties used are defined as follows: Young's modulus = 193.1×10^9, Poisson's ratio = 0.3, and density = 785.

In the first step a nonuniform body force of 1.0×10^5 is applied in the x-direction for all the elements except the axisymmetric elements, where it is applied in the r-direction. The amplitude function for this nonuniform body force is defined such that the load is ramped on over the first half of the step and held constant for the rest of the analysis. In the second step another nonuniform body force of 1.0×10^5 is applied in the y-direction for all the elements except the axisymmetric elements, where it is applied in the z-direction. This load is applied using the same amplitude function that was used in the first step. For C3D4, C3D6, C3D8R, S3R, S4R, M3D3, and M3D4R elements, another nonuniform body force of 1.0×10^5 is applied in the z-direction in a third step. This load also has the same amplitude function that was used in the first step.

Results and discussion

The results for all of the elements agree with the analytical values, which are included at the top of the input file.

Input files

element_nbody.inp	Input data used for this test.
element_nbody.f	User subroutine used for this test.

Nonuniform pressure load

Elements tested

Distributed element-based loads

B21 B31 PIPE21 PIPE31 CPE3 CPE4R CPS3 CPS4R CAX3 CAX4R C3D4 C3D6 C3D8R M3D3 M3D4R R2D2 RAX2 R3D3 R3D4 SAX1 S3R S4R

Distributed surface loads

CPE3 CPE4R CPS3 CPS4R CAX3 CAX4R C3D4 C3D6 C3D8R M3D3 M3D4R R2D2 RAX2 R3D3 R3D4 SAX1 S3R S4R

Features tested

Nonuniform pressure load prescribed with distributed element-based loads and distributed surface loads.

Problem description

In this verification test all the available element types are tested by loading them with a nonuniform pressure load. All the element nodes are fixed in position, and the reaction forces generated at the nodes are used to verify the element load calculations. The purpose of this test is to verify the element load calculations, not to test all the capabilities of user subroutine *VDLOAD*.

The material model is isotropic linear elasticity. The material properties used are defined as follows: Young's modulus = 193.1×10^9, Poisson's ratio = 0.3, and density = 785.

In the first step a nonuniform pressure of 1.0×10^5 is applied on element edges (for CPE3, CPE4R, CPS3, CPS4R, CAX3, CAX4R, SAX1, R2D2, and RAX2 elements) or element faces (for C3D4, C3D6, C3D8R, S3R, S4R, M3D3, M3D4R, R3D3, and R3D4 elements). For the beam and pipe elements (B21, B31, PIPE21, and PIPE31) a nonuniform force per unit length of 1.0×10^5 is applied in the y-direction. The amplitude function for this nonuniform pressure load is defined such that the load is ramped on over the first half of the step and held constant for the rest of the analysis. In the second step a nonuniform pressure of 1.0×10^5 is applied on a different element edge (for CPE3, CPE4R, CPS3, CPS4R, CAX3, and CAX4R elements) or element face (for C3D4, C3D6, and C3D8R elements). For the beam elements (B21 and B31) a nonuniform force per unit length of 1.0×10^5 is applied in the x-direction. These loads are applied using the same amplitude function that was used in the first step.

Results and discussion

The results for all of the elements agree with the analytical values, which are included at the top of the input file.

Input files

element_npres.inp	Input data for element-based loads used for this test.
surface_npres.inp	Input data for surface-based loads used for this test.
element_npres.f	User subroutine used for this test.

VFILM

Products: Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D8RT CPS4RT S4RT

Features tested

User subroutine *VFILM* to define a film coefficient and sink temperature for an Abaqus/Explicit heat transfer analysis.

Problem description

The tests verify that the user-defined nonuniform film coefficient and sink temperature in user subroutine *VFILM* transferred properly to the solution process. The finite element models consist of element-based (**FILM*), surface-based (**SFILM*), and node-based (**CFILM*) film convection loads. The film coefficient and sink temperature are implemented as user-defined values. For comparison purposes, a duplicate set of tests with the film coefficient and sink temperature directly defined in input files is also included to provide reference solutions.

Results and discussion

In each case the results for the cases using user subroutine *VFILM* match the solutions in the reference tests.

Input files

film_ele_c3d8rt_vfilm.inp	Element-based film load test using user subroutine *VFILM*.
film_ele_c3d8rt_vfilm.f	User subroutine used in `film_ele_c3d8rt_vfilm.inp`.
film_ele_c3d8rt.inp	Element-based film load test without using user subroutine *VFILM*.
sfilm_s4rt_c3d8rt_vfilm.inp	Surface-based film load test using user subroutine *VFILM*.
sfilm_s4rt_c3d8rt_vfilm.f	User subroutine used in `sfilm_s4rt_c3d8rt_vfilm.inp`.
sfilm_s4rt_c3d8rt.inp	Surface-based film load test without using user subroutine *VFILM*.
cfilm_cflux_cps4rt_vfilm.inp	Node-based film load test using user subroutine *VFILM*.
cfilm_cflux_cps4rt_vfilm.f	User subroutine used in `cfilm_cflux_cps4rt_vfilm.inp`.
cfilm_cflux_cps4rt.inp	Node-based film load test without using user subroutine *VFILM*.

VFRIC, VFRIC_COEF, and VFRICTION

Products: Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

User subroutines testing static and dynamic friction in a stress/displacement analysis

Elements tested

CPE3 C3D8R MASS

Features tested

User subroutines *VFRIC*, *VFRIC_COEF*, and *VFRICTION* to define frictional behavior.

Problem description

The problems in this section demonstrate modeling of frictional behavior with user subroutines *VFRIC*, *VFRIC_COEF*, and *VFRICTION*.

The first example uses user subroutines *VFRIC*, *VFRIC_COEF*, and *VFRICTION* that are coded with the Coulomb model for frictional behavior, which is also the default model in Abaqus. The critical shear stress, τ_{crit}, at which surfaces begin to slide with respect to each other is given as

$$\tau_{crit} = \mu p,$$

where μ is the coefficient of friction and p is normal pressure.

The second example uses user subroutines *VFRIC*, *VFRIC_COEF*, and *VFRICTION* for rate-dependent Coulomb friction behavior where the evolution of the coefficient of friction, μ, is given by an exponential law

$$\mu = \mu_k + (\mu_s - \mu_k)\, e^{-d_c \dot{\gamma}_{eq}},$$

where μ_s is the static coefficient of friction, μ_k is the kinetic coefficient of friction, d_c is the decay coefficient, and $\dot{\gamma}_{eq}$ is the magnitude of the tangential slip velocity.

Both friction models are tested on a mesh of a rectangular block (length 5 in, height 1 in, and depth 1 in, elastic modulus 30×10^6 psi, density 7.3×10^{-4} lbf s^2/in^4) of two CPE3 or C3D8R elements sliding over a flat analytical rigid surface along its length in the x-direction. A uniform pressure of 2000 psi is applied on the top face of the block, and an initial velocity of 200 in/s is prescribed at each node on the block. The same problem is used to test the friction models provided in Abaqus/Explicit in *Friction models in Abaqus/Explicit*.

For the Coulomb model $\mu =$ 0.15; for the rate-dependent Coulomb model $\mu_s =$ 0.15, $\mu_k =$ 0.05, and $d_c =$ 0.01 s/in.

Results and discussion

The results for the two models are discussed below.

Results for the default Coulomb model

The prescribed external load gives a normal pressure of 2000 psi and a frictional stress of 300 psi. This corresponds to a negative acceleration of 4.109589×10^5 in/s^2 in the tangential direction since the frictional stress opposes the motion of the block. Given the initial velocity and the acceleration, the block should come to rest after sliding over a distance of 4.866×10^{-2} in over a time period of 4.866×10^{-4} s. The corresponding values of sliding distance and time period for the finite element model with user subroutines are 4.866×10^{-2} in and 4.878×10^{-4} s, respectively. The numerical results show some oscillations in the normal reactions and frictional forces caused by the inertial effect of nodes on the top of the block; even after the block stops sliding, there is some oscillation of the block in a shear mode.

Results for the rate-dependent Coulomb model

In this model the velocity of the node in contact corresponds to the slip rate for the friction model. To verify the friction model, we compare the velocity values obtained using the analytical expression with the average velocity values of the nodes in contact obtained from the finite element model with user subroutines (see *Table 1* and *Table 2*). Small differences occur between the analytical and numerical values of velocity because of small oscillations in a shear mode in the finite element model. The analysis using penalty contact with user subroutines *VFRIC* and *VFRIC_COEF* has additional differences due to default viscous contact damping, which contributes to the contact forces opposing the motion of the block.

Table 1: Comparison of velocity values for the rate-dependent Coulomb model for user subroutine VFRIC.

Time	Velocity	Velocity
10^{-4} s	(Analytical) in/s	(*VFRIC*) in/s
1.0301	181.7	181.8
2.0042	163.6	164.2
3.0001	144.1	143.5
4.0064	123.1	123.9
5.0000	100.6	100.6
6.0284	74.73	75.19
7.0022	46.87	47.97
8.0017	12.88	11.89
8.2289	4.054	2.965

Table 2: Comparison of velocity values for the rate-dependent Coulomb model for user subroutines VFRIC_COEF and VFRICTION.

Time	Velocity	Velocity
10^{-4} s	(*VFRIC_COEF*) in/s	(*VFRICTION*) in/s
1.0142	183.0	183.0
2.0145	164.7	164.7
3.0159	143.7	143.7
4.0162	121.0	121.0
5.0000	99.92	99.95
6.0146	77.09	77.12

Time	Velocity	Velocity
10^{-4} s	(*VFRIC_COEF*) in/s	(*VFRICTION*) in/s
7.0149	46.92	46.98
8.0156	10.90	11.00
8.2289	2.85	4.69

Input files

vfric_coul.inp	Input data that refer to user subroutine *VFRIC* with the Coulomb model.
vfric_coul.f	User subroutine *VFRIC* for the Coulomb model.
vfric_coul_part1.inp	Input data (with the model defined in terms of an assembly of part instances) that refer to user subroutine *VFRIC* with the Coulomb model and the utility routine VGETPARTINFO.
vfric_coul_part1.f	User subroutine *VFRIC* for the Coulomb model that illustrates the use of the utility routine VGETPARTINFO.
vfric_coul_part2.inp	Input data (with the model defined in terms of an assembly of part instances) that refer to user subroutine *VFRIC* with the Coulomb model and the utility routine VGETINTERNAL.
vfric_coul_part2.f	User subroutine *VFRIC* for the Coulomb model that illustrates the use of utility routine VGETINTERNAL.
vfric_rdcoul.inp	Input data that refer to user subroutine *VFRIC* with the rate-dependent Coulomb model.
vfric_rdcoul.f	User subroutine *VFRIC* for the rate-dependent Coulomb model.
vfric_rdcoulpnlty.inp	Input data that refer to user subroutine *VFRIC* with the rate-dependent Coulomb model and penalty contact.
vfric_coef_coul.inp	Input data that refer to user subroutine *VFRIC_COEF* with the Coulomb model.
vfric_coef_coul.f	User subroutine *VFRIC_COEF* for the Coulomb model.
vfriction_coul.inp	Input data that refer to user subroutine *VFRICTION* with the Coulomb model.
vfriction_coul.f	User subroutine *VFRICTION* for the Coulomb model.
vfric_coef_rdcoul.inp	Input data that refer to user subroutine *VFRIC_COEF* with the rate-dependent Coulomb model.
vfric_coef_rdcoul.f	User subroutine *VFRIC_COEF* for the rate-dependent Coulomb model.
vfriction_rdcoul.inp	Input data that refer to user subroutine *VFRICTION* with the rate-dependent Coulomb model.
vfriction_rdcoul.f	User subroutine *VFRICTION* for the rate-dependent Coulomb model.

User subroutine tested in a coupled temperature-displacement analysis

Elements tested

C3D8RT

Features tested

User subroutine to define frictional behavior for contact surfaces in a coupled temperature-displacement analysis.

Problem description

The problem described in Part II of *FRIC* is solved using Abaqus/Explicit. A transient analysis is performed. The mechanical and thermal properties are identical to those used in the analysis performed with Abaqus/Standard. Only two steps are required for the Abaqus/Explicit simulation: a downward force is applied in the first step to establish and maintain contact between the blocks, and a tangential force is applied in the second step to promote sliding between the blocks. In each step the mechanical and thermal loads are applied gradually to ensure a quasi-static response. The total applied tangential force is 0.18 (versus 100 in Abaqus/Standard); this is the force required to generate a total slip of 0.15 over a time interval of 1000 when the load is prescribed with a ramp function.

Results and discussion

The results obtained with Abaqus/Explicit compare well with the analytical solution for the total slip (the total slip predicted by Abaqus/Explicit is 0.145). Closer agreement with the analytical solution can be obtained by reducing the loading rate. This further reduces the effects of material inertia on the response.

Input files

vfric_c3d8rt.inp	Coupled temperature-displacement analysis.
vfric_c3d8rt.f	User subroutine for the coupled temperature-displacement analysis.

VHETVAL

Products: Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

CPS4RT

Features tested

User subroutine to provide internal heat generation in a material.

Problem description

This test verifies that the user-defined internal generated heat flux in user subroutine *VHETVAL* is transferred properly to the solution process: the energy balance is satisfied. The finite element model consists of two disconnected 2D rectangular blocks of material, 10×2. The testing material block has heat generated internally within its volume by user subroutine *VHETVAL*. The value of the generated heat flux, $r = 0.40483$, is defined and is passed into the user subroutine via the `PROPS` array. The reference material block is heated by an external volumetric heat flux with a value of $r = 0.40483$. The materials have specific heat, $c = 0.1431$, and density, $\rho = 0.2829$. Dummy mechanical properties are prescribed to complete the material definition.

Based on the energy balance, a transient thermal analysis with all edges of the volume insulated should give a temperature rate of

$$\frac{d\theta}{dt} = \frac{r}{\rho c} = 10.$$

Results and discussion

The initial nodal temperatures are zero. Time is incremented by 0.961 units in each increment of the analysis. From the equation above, nodal temperatures should increment by 9.6 units during each increment. The nodal temperatures of the testing material block with *VHETVAL* also match the results of the reference material block subject to an external thermal load.

Increment	Time	Temperature
1	0.96	9.61
10	9.61	96.1
20	19.22	192.2
30	28.83	288.3
40	38.44	384.4
50	47.57	475.7
53	50.00	500.0

Input files

vhetvalx.inp	Input file for this analysis.
vhetvalx.f	User subroutine *VHETVAL* used in vhetvalx.inp.

VSDVINI

Products: Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

S4R

Features tested

User subroutine to define initial conditions for solution-dependent state variables.

Problem description

Two test cases are included. The purpose of both tests is to show that the initial conditions of the solution-dependent state variables are interpreted correctly and to show that the solution-dependent state variables can be updated in another user subroutine.

In the first test case six solution-dependent state variables are initialized in user subroutine *VSDVINI* and are subsequently updated in user subroutine *VUMAT*. The problem is a trivial dynamic analysis of a single shell element with a linear elastic material.

In the second test case two solution-dependent state variables are initialized in user subroutine *VSDVINI* and are assigned to two user-defined field variables used to evaluate the elastic properties of the material.

Results and discussion

The solution-dependent state variables are correctly initialized in *VSDVINI*. They are made available properly in *VUMAT* in the first test case, and their values are assigned correctly to the two field variables in the second test case.

Input files

vsdvini_vumat_s4r.inp	Input file for the first test case of *VUMAT* analysis.
vsdvini_vumat_s4r.f	User subroutine *VSDVINI* used in vsdvini_vumat_s4r.inp.
vsdvini_s4r_fv.inp	Input file for the second test case.
vsdvini_s4r_fv.f	User subroutine *VSDVINI* used in vsdvini_s4r_fv.inp.

VUAMP

Products: Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.



Warning:

User subroutine *VUAMP* provides the user with a very general option to interface with the code. With any use of this subroutine interface, extensive verification should be done to make sure that the results are correct.

Features tested

User subroutine *VUAMP* to define amplitudes.

Problem description

The finite element models for most test cases consist of simple linear truss or connector elements. User subroutine *VUAMP* is used to define amplitudes that are subsequently used to drive certain loading options such as concentrated loads, boundary conditions, and connector motions. In most cases the *VUAMP* user-defined amplitudes are simple linear ramps. The results from the analyses are compared against reference results obtained using identical models with equivalent tabular amplitude definitions.

User subroutine *VUAMP* can make use of sensor definitions and state variables, and a number of tests exercise these features. In certain tests (such as when a user-defined amplitude is used to drive the displacement boundary condition) the user subroutine may compute derivatives, integrals, and second derivatives of the amplitude function being defined. In addition, amplitude properties can be defined on the data line, which allows changes to the amplitude definition without modifing the user subroutine.

Results and discussion

The verification consists of comparing the results obtained from the model using user-defined amplitudes with the corresponding model using tabular amplitudes. The results match very well, as expected.

Input files

vuamp_ramp_simple_cload.inp	Concentrated load force scaled by an amplitude defined in user subroutine *VUAMP* in vuamp_ramp_simple.f.
vuamp_ramp_simple_bcdisp.inp	Displacement boundary condition scaled by an amplitude defined in user subroutine *VUAMP* in vuamp_ramp_simple.f.
vuamp_ramp_simple_restart.inp	Displacement boundary condition in a restart analysis scaled by an amplitude defined in user subroutine *VUAMP* in vuamp_ramp_simple.f.
vuamp_ramp_simple_bcvel.inp	Velocity boundary condition scaled by an amplitude defined in user subroutine *VUAMP* in vuamp_ramp_simple.f.
vuamp_ramp_simple_bcvel_props.inp	Velocity boundary condition scaled by an amplitude defined in user subroutine *VUAMP* in vuamp_ramp_simple_props.f.

vuamp_ramp_simple_bcacc.inp	Acceleration boundary condition scaled by an amplitude defined in user subroutine *VUAMP* in vuamp_ramp_simple.f.
vuamp_ramp_simple_connmot.inp	Displacement connector motion scaled by an amplitude defined in user subroutine *VUAMP* in vuamp_ramp_simple.f.
vuamp_ramp_state_connmot_ props.inp	Displacement connector motion scaled by an amplitude defined in user subroutine *VUAMP* in vuamp_ramp_simple_props.f.
vuamp_ramp_simple.f	User subroutine *VUAMP* defining a simple ramp amplitude using state variables.
vuamp_ramp_simple_props.f	User subroutine *VUAMP* defining a simple ramp amplitude using state variables. The amplitude property, as defined, acts as the slope of the ramp function.
vuamp_ramp_state_cload.inp	Concentrated load force scaled by an amplitude defined in user subroutine *VUAMP* in vuamp_ramp_state.f.
vuamp_ramp_state_connmot.inp	Displacement connector motion scaled by an amplitude defined in user subroutine *VUAMP* in vuamp_ramp_state.f.
vuamp_ramp_state.f	User subroutine *VUAMP* defining a simple ramp amplitude using state variables.
vuamp_ramp_state_props.f	User subroutine *VUAMP* defining a simple ramp amplitude using state variables. The amplitude property, as defined, acts as the slope of the ramp function.
vuamp_deriv_bcdisp.inp	Displacement boundary condition scaled by an amplitude defined in user subroutine *VUAMP* in vuamp_deriv.f.
vuamp_deriv.f	User subroutine *VUAMP* defining cubic function amplitude which also computes derivatives.
vuamp_ramp_sensor_connmot.inp	Displacement connector motion scaled by an amplitude defined in user subroutine *VUAMP* in vuamp_ramp_sensor.f.
vuamp_ramp_sensor.f	User subroutine *VUAMP* defining a simple ramp amplitude using sensors.
vuamp_manysensors.inp	Displacement connector motion scaled by an amplitude defined in user subroutine *VUAMP* in vuamp_manysensors.f.
vuamp_manysensors.f	User subroutine *VUAMP* defining a simple ramp amplitude using sensors.

VUCHARLENGTH

Products: Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D8 C3D8R CAX4R CPE4 CPE4R CPS4R
M3D4 M3D4R S4 S4R S4RS SAX1 SC8R T2D2 T3D2

Features tested

User subroutine to define the characteristic element length to be used by Abaqus for the regularization of models that exhibit strain softening or to be passed to user subroutines that are called at the material point.

Problem description

This test verifies that the characteristic element length defined in user subroutine *VUCHARLENGTH* is transferred properly to built-in Abaqus material models and user subroutines that are called at the material point. The finite element model consists of multiple disconnected elements of the types listed. Each element is associated with a Mises plasticity model, and in each case a damage model is constructed using user subroutine *VUSDFLD*. For comparison purposes a duplicate set of elements with equivalent built-in Abaqus damage initiation/damage evolution models are included to provide a reference solution. Both the testing element and the reference element use the user-defined characteristic element length for the damage analysis. The user-defined characteristic element length is specified to be a function of nodal coordinates and the shape of the element.

Results and discussion

The initial value of the characteristic element length can be evaluated by hand calculation. The characteristic element length is stored as a solution-dependent variable in user subroutine *VUSDFLD*. The value of the characteristic element length in the second increment, observed in the time-history plots of solution-dependent variables, agrees with the value obtained from hand calculation. The time history plots of the other output variables in the testing element match the solution in the reference element.

Input files

damage_vucharlength.inp	Damage analysis.
damage_vucharlength.f	User subroutines *VUCHARLENGTH* and *VUSDFLD* used in damage_vucharlength.inp.

VUCREEPNETWORK

Products: Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D8

Features tested

User subroutine to define time-dependent behavior for models defined within the parallel rheological framework.

Problem description

The input file consists of two elements with the material response specified using the parallel rheological framework with two networks using the power law strain hardening and hyperbolic-sine creep laws. The same material models are assigned to each element; however, the viscous behavior in the networks is defined in two different ways. In one case user subroutine *VUCREEPNETWORK* is used to specify the power law strain hardening and hyperbolic-sine creep law. In the second case the built-in Abaqus creep laws are used. The analysis consists of one step, and both elements are subjected to the same load history. The specimens are loaded rapidly at the beginning of the step and the load is maintained throughout the step, so the materials are subjected to creep.

Results and discussion

The results obtained using the two equivalent materials show very good agreement.

Input files

x_viscnet_c3d8_usercreep.inp	The two-element model with two network viscoelastic materials.
x_viscnet_vucreepnetwork.f	User subroutine to define the power law strain hardening and hyperbolic-sine creep laws.

VUEL

Products: Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.



Warning:

User subroutine *VUEL* provides the user with a very general option to interface with the code. With any use of this subroutine interface, extensive verification should be done to make sure that the results are correct.

Elements tested

CONN3D2 T3D2

Features tested

User subroutine *VUEL* to define the element mass matrix, the force vector, and the stable time increment.

Problem description

The finite element model for most test cases consists of linear truss or spring elements defined using user subroutine *VUEL*. In most cases the results from the analysis are compared against reference results obtained using an identical model with Abaqus elements T3D2 and CONN3D2. A truss element lying along the global *X*-axis is defined in a user subroutine. This user element is used in a model with single or multiple user elements subjected to concentrated loads. The results are then compared with models using T3D2 elements.

The field and temperature-dependent variation in material properties can be defined in the user element. To test this capability, a truss element is defined with a linear variation of elastic modulus with temperature and field variables. The results are then compared with temperature and field variable dependencies in properties in a T3D2 element.

A number of uncoupled linear spring elements are defined using the user subroutine interface. Both two-dimensional and three-dimensional elements are tested. The following features are tested with three-dimensional spring elements: an element with more than two nodes, an element with degrees of freedom ordered in a nonstandard way, and an element with a different number of degrees of freedom at its nodes.

User-defined elements with acoustic degrees of freedom and with heat transfer capabilities are also tested.

Results and discussion

The verification consists of comparing the results obtained from the model using user-defined elements with the corresponding model using regular Abaqus elements. The results are as expected.

Input files

vuel_truss_3d_1el.inp	Analysis of a one-element truss defined in user subroutine *VUEL* in vuel_truss.f.

vuel_truss_3d_1el_dload.inp	Analysis of a one-element truss defined in user subroutine *VUEL* in vuel_truss.f using user-coded external loads.
vuel_truss_3d_250el.inp	Analysis of multiple truss elements connected in series; the truss element is defined in user subroutine *VUEL* in vuel_truss.f.
vuel_truss.f	User subroutine *VUEL* defining a three-dimensional truss element along the global *x*-direction.
abq_truss_3d_1el.inp	Analysis of a one-element truss defined using the T3D2 element; reference solution for vuel_truss_3d_1el.inp.
abq_truss_3d_250el.inp	Analysis of multiple truss elements defined using the T3D2 element; reference solution for vuel_truss_3d_250el.inp.
vuel_truss_3d_1el_fieldvar.inp	A one-element truss with field variable-dependent material properties. The element is defined in user subroutine *VUEL* in vuel_truss_fieldvar.f.
vuel_truss_fieldvar.f	User subroutine *VUEL* defining a three-dimensional truss element with field variable-dependent material properties.
abq_truss_3d_1el_fieldvar.inp	A T3D2 element with field variable-dependent material properties; reference solution for vuel_truss_3d_1el_fieldvar.inp.
vuel_truss_3d_1el_temp.inp	A one-element truss with temperature-dependent material properties. The element is defined in user subroutine *VUEL* in vuel_truss_temp.f.
vuel_truss_temp.f	User subroutine *VUEL* defining a three-dimensional truss element with temperature-dependent material properties.
abq_truss_3d_1el_temp.inp	A T3D2 element with temperature-dependent material properties; reference solution for vuel_truss_3d_1el_temp.inp.
vuel_springs_3d_freevibx.inp	Free vibration in the global *x*-direction of a linear spring defined using user subroutine *VUEL* in vuel_springs_3d.f.
abq_connector_freevibx.inp	Free vibration in the global *x*-direction of a linear spring defined using connector element CONN3D2; reference solution for vuel_springs_3d_freevibx.inp.
vuel_springs_3d_freevibrotx.inp	Free rotational vibration about the global *x*-direction of a linear spring defined using user subroutine *VUEL* in vuel_springs_3d.f.
abq_connector_freevibrotx.inp	Free rotational vibration about the global *x*-direction of a linear spring defined using connector element CONN3D2; reference solution for vuel_springs_3d_freevibrotx.inp.
vuel_springs_3d_1el.inp	A one-element linear spring subjected to tensile loading; the element is defined using user subroutine *VUEL* in vuel_springs_3d.f.
vuel_springs_3d_10el.inp	Multiple linear springs subjected to tensile loading; the element is defined using user subroutine *VUEL* in vuel_springs_3d.f.
vuel_springs_3d.f	User subroutine *VUEL* defining a three-dimensional linear spring element.
vuel_springs_3d_3node.inp	A one-element linear spring with three nodes subjected to tensile loading; the element is defined using user subroutine *VUEL* in vuel_springs_3d_3node.f.
vuel_springs_3d_3node.f	User subroutine *VUEL* defining a three-dimensional linear spring element with three nodes.
vuel_springs_3d_diffDofs.inp	A one-element linear spring with a different number of degrees of freedom defined at its nodes—the first node has six degrees of freedom, whereas the second node has only three degrees of freedom; the element is defined in user subroutine *VUEL* in vuel_springs_3d_diffDofs.f.
vuel_springs_3d_diffDofs.f	User subroutine *VUEL* defining a three-dimensional spring element with a different number of degrees of freedom at its two nodes.

vuel_springs_3d_jumpDofs.inp	A one-element linear spring with degrees of freedom ordered in a nonstandard way; the first degree of freedom from all the nodes is listed first, followed by second degree of freedom from all the nodes, and so on.
vuel_springs_3d_jumpDofs.f	User subroutine *VUEL* defining a three-dimensional spring element with degrees of freedom ordered in a nonstandard way.
vuel_springs_2d_1el.inp	A one-element two-dimensional linear spring. The element is defined in user subroutine *VUEL* in vuel_springs_2d.f.
vuel_springs_2d.f	User subroutine *VUEL* defining a two-dimensional linear spring element.
vuel_acoustic_1el.inp	A one-element input file to test acoustic user elements defined in user subroutine *VUEL* in vuel_acoustic.f.
vuel_acoustic.f	User subroutine *VUEL* defining an acoustic element.
vuel_heat_transf_1el.inp	A one-element input file to test a heat transfer user element defined in user subroutine *VUEL* in vuel_heat_transf.f.
vuel_heat_transf.f	User subroutine *VUEL* defining a heat transfer element.

VUFIELD

Products: Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

CONN3D2 CPE4R SPRINGA

Features tested

User subroutine to define field variable values.

Problem description

Three tests are provided to demonstrate the use of user subroutine *VUFIELD* and to verify that field variable values are properly transferred to a structure when the values are set using user subroutines.

The first example tests connector behavior by specifying field variables through both input file and user subroutine *VUFIELD*. The field variable value specified in *VUFIELD* is constant and applies to the second field variable. The value for the first field variable is defined in the input file using an amplitude.

In the second example the number of nodes passed into user subroutine *VUFIELD* is limited to the number specified by the blocking size. The amplitude specified in the data line does not affect field variable values defined in the user subroutine.

The third example defines two field variables on the given node set. These field variables are passed into user subroutine *VUFIELD* at the same time, so their values can be updated simultaneously. The returned field variable values are then modulated by an amplitude.

Results and discussion

Results obtained from using user subroutine *VUFIELD* are compared with those where field variable values are specified directly in the data line. The results match.

Input files

connectorbehv_vufield.inp	Input file for the first example.
connectorbehv_vufield.f	User subroutine used in connectorbehv_vufield.inp.
rigidwithfric_vufield.inp	Input file for the second example.
rigidwithfric_vufield.f	User subroutine used in rigidwithfric_vufield.inp.
springs_vufield.inp	Input file for the third example.
springs_vufield.f	User subroutine used in springs_vufield.inp.

VUGENS

Products: Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

S4R

Features tested

User subroutine to define the mechanical behavior of a shell section in terms of generalized section quantities. The subroutine returns the section forces and moments as well as the section stiffness matrix of the element.

Problem description

To verify user subroutine *VUGENS*, the data line input option is used to specify the shell section stiffness and thickness of the shell elements (passed into *VUGENS* by array `PROPS`).

ver_prystructdcoup_vugens.inp

S4R elements with user subroutine *VUGENS* are used to model a pair of surfaces connected by a spot weld using the **FASTENER* definition. A fixed boundary condition is applied to the nodes of one surface, and concentrated forces are applied to the nodes of the other surface. For comparison purposes, a duplicate pair of surfaces modeled by S4R elements without using user subroutine *VUGENS* is included to provide a reference solution.

Results and discussion

The verification compares results between models using a general shell section with and without user subroutine *VUGENS*. In each case, the results were identical.

Input files

ver_prystructdcoup_vugens.inp	Model using S4R elements with user subroutine *VUGENS*.
ver_prystructdcoup_vugens.f	User subroutine *VUGENS* used in `ver_prystructdcoup_vugens.inp`.

VUHARD

Products: Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D8R

Features tested

User subroutine to define hardening for the following plasticity models: Mises plasticity, Hill plasticity, combined hardening, and porous metal plasticity.

Problem description

This test verifies that the user-defined yield stress and its derivatives in user subroutine *VUHARD* are transferred properly to the solution process. The finite element model consists of multiple disconnected cubes made of a single C3D8R element. Each element is associated with one of the plasticity models listed above, and in each case a user-defined hardening is implemented. For comparison purposes, a duplicate set of elements with equivalent plasticity and isotropic hardening definitions is included to provide a reference solution.

Results and discussion

In each case the results in the testing element match the solution in the reference element.

Input files

iso_harden_vuhard.inp	Plasticity model.
iso_harden_vuhard.f	User subroutine *VUHARD* used in iso_harden_vuhard.inp.

VUHYPER

Products: Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Features tested

User subroutine to define Mooney-Rivlin and neo-Hookean hyperelastic material properties.

User subroutine for hyperelastic model

Elements tested

C3D8R CPS4R

Problem description

This set of verification problems tests the variables that are passed into user subroutine *VUHYPER*. In each test the material properties are specified using general hyperelastic material properties for the first element and the strain energy function and the associated derivatives defined in user subroutine *VUHYPER* for the second element.

Two different sets of material data are used, as described below.

Material 1:

Polynomial (N=1), compressible.
C_{10} = 80.0.
C_{01} = 20.0.
D_1 = 1.0×10^{-3}.

Material 2:

Polynomial (N=1), compressible, field variable dependency included for user subroutine *VUHYPER*.
C_{10} = 80.0.
C_{01} = 20.0.
D_1 = 1.5×10^{-3}.

Results and discussion

Since the same boundary conditions are applied on the corresponding nodes of the two elements, the results are expected to be identical in the two cases, thus verifying the use of user subroutine *VUHYPER* with the second element.

Input files

vuhypercp2s.inp	Compressible, biaxial tension, 2D, state variables, Material 1.
vuhypercp2s.f	User subroutine *VUHYPER* used in `vuhypercp2s.inp`.
vuhyperct3f.inp	Compressible, uniaxial tension, 3D, field variables, Material 2.
vuhyperct3f.f	User subroutine *VUHYPER* used in `vuhyperct3f.inp`.

User subroutine for defining different hyperelastic models within the parallel rheological framework

Elements tested

C3D8R

Problem description

The tests in this section define different hyperelastic material models in different networks of a nonlinear viscoelastic material model using the parallel rheological framework (see *Parallel Rheological Framework* and **VISCOELASTIC*). For the purpose of verification, the results are compared with those obtained using an element overlay approach. The element overlay consists of two elements sharing the same nodes. The first element of the overlay has material properties defined by a polynomial hyperelastic material model (Material 3), while the second element of the overlay uses a neo-Hookean hyperelastic model with a nonlinear viscoelastic network (Material 4), defined such that there is no contribution from the equilibrium network. A third element is introduced to test user subroutine *VUHYPER* in combination with a parallel rheological framework definition using the properties described in Material 5. The properties and stiffness ratios of Material 5 are defined such as to provide the exact same response as the overlay of Materials 3 and 4. The user subroutine defines the strain energy density function with a polynomial hyperelastic model for the equilibrium network (*networkid*=0) and with a neo-Hookean strain energy function for the nonlinear viscoelastic network, (*networkid*=1), as described in Material 5. All elements are subjected to the same uniaxial tension loading and the same boundary conditions.

Material 3:

Polynomial (*N*=1), compressible.
C_{10} = 40.0.
C_{01} = 10.0.
D_1 = 2.0×10^{-3}.

Material 4:

Neo-Hookean, compressible, with one nonlinear viscoelastic branch defined by hyperbolic-sine creep model.	
Network stiffness ratio, r_1= 1.0 (r_0 = 0.0).	
The coefficients of neo-Hookean hyperelastic model are:	
	C_{10} = 25.0.
	D_1 = 2.0×10^{-3}.
The coefficients of Hyperbolic-sine creep model are:	
	A = 2.5×10^{-5}.

	B = 0.3.
	n = 1.0.

Material 5:

User-defined hyperelastic material.	
Network stiffness ratio, r_1 = 0.5 (r_0 = 0.5).	
C_{10} = 80.0.	
C_{01} = 20.0.	
D_1 = 1.0 × 10^{-3}.	
The user-defined hyperelastic properties of network 1 (N=1) are:	
	C_{10} = 50.0.
	D_1 = 1.0 × 10^{-3}.
The coefficients of hyperbolic-sine creep model in network 1 are:	
	A = 2.5 × 10^{-5}.
	B = 0.15.
	n = 1.0.

Results and discussion

Since the same boundary conditions are applied on the corresponding nodes of all the three elements, the reaction forces are the same for the nodes of the overlayed elements and the nodes of the element using the parallel rheological framework in combination with user subroutine *VUHYPER*. In addition, the sum of the stress components in the overlayed elements is equal to the corresponding stress components in the *VUHYPER* element.

Input files

vuhyper_tension_prf_twomaterials.inp	Element overlay, compressible, uniaxial tension, 3D, Material 3 and Material 4.
vuhyper_tension_prf_twomaterials.f	User subroutine *VUHYPER* used in `vuhyper_tension_prf_twomaterials.inp`.

VUHYPER_STRETCH

Products: Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Features tested

User subroutine to define Hencky material properties.

Elements tested

C3D8

Results and discussion

Since the same boundary conditions are applied on the corresponding nodes of the two elements, the results are expected to be identical. The match verifies the use of user subroutine *VUHYPER_STRETCH*.

Input files

xpl_userhyperstretch_shear_c3d8.inp	Compressible, shear, 3D, Material 1.
vuhyper_stretchc3d8.f	User subroutine *VUHYPER_STRETCH* used in `xpl_userhyperstretch_shear_c3d8.inp`.

VUINTER

Products: Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Features tested

User subroutine to define contact interface behavior.

Problem description

User subroutine *VUINTER* used in this example models a mechanically compliant, thermally conductive contact interface material with uniform thickness. The interface material is assumed to be bonded to each of two contacting surfaces. The interface material exhibits elastic-plastic behavior with linear hardening in the normal contact direction and purely elastic resistance to relative sliding. Membrane straining of the interface does not affect the stress transmitted to the surfaces. The interface material is thermally conductive; and the conductance remains constant, independent of the gap or pressure at the interface.

With this interface model all secondary nodes within a specified initial gap distance relative to the main surface remain bonded throughout the analysis. The other secondary nodes are not bonded (they never have contact forces or heat fluxes applied). The initial gap (see the `gapInit` variable in the example user subroutine) accounts for surface thicknesses (equal to zero in these examples) as well as the specified interface thickness. It is assumed that the initial strain of the interface is zero. Abaqus/Explicit will not make strain-free adjustments to resolve initial overclosures or gaps for contact pairs that use user subroutine *VUINTER*.

Specification of the interface thickness is optional; it is used here for convenience so that the interface thickness will be used when calculating the penetration or gap for each node (the variable `rdisp(1,...)`). Alternatively, you could model an interface thickness within the user subroutine by constructing a state variable that contains an offset value for each node. This offset can be a function of the initial penetration and the interface thickness at the node (for example, set the offset equal to the negative of the initial penetration). The actual penetration would then be the sum of the value given in `rdisp(1,...)` and the stored nodal offset value. Since strain-free adjustments are not made to the nodes, this procedure allows a convenient way to eliminate any spurious initial contact forces resulting from inaccurate nodal coordinates, removing the requirement to position the surface nodes accurately when constructing a model.

Strain increments in the normal direction are calculated within the user subroutine as the change in contact penetration divided by a specified interface thickness. This thickness is a property of the interface model. For consistency, this thickness should be set to the same value as the interface thickness in these examples. Strain increment components corresponding to transverse shearing of the interface are likewise computed as the appropriate sliding increment component divided by the specified interface thickness.

Heat fluxes are calculated by multiplying the thermal conductivity of the interface material by the nodal area and temperature difference between the secondary node and main contact point and dividing by the initial interface thickness. Effects such as heat generation due to friction are not taken into account.

A complete list of properties specified for this interface model, in the order in which the values are specified on the second data line of the contact property definition, is as follows:

1. Gap cutoff distance. Only secondary nodes with initial gaps less than this distance are bonded.
2. Young's modulus of the interface material.

3. Poisson's ratio of the interface material.
4. Initial yield stress in the normal direction of the interface material.
5. Hardening modulus in the normal direction of the interface material.
6. Interface thickness used in the strain calculations.
7. Thermal conductivity (units of $\mathbf{J\theta^{-1}T^{-1}L^{-1}}$) of the interface material used for contact heat flux calculations.

Three user-defined state variables are employed in this example. The first simply indicates whether the initialization to determine which nodes are bonded has been completed. The second is used to mark which nodes are bonded. The third keeps track of the current yield stress at each secondary node.

Two simple configurations are used to test this user subroutine in both two and three dimensions. In the first configuration each of two identical elastic bodies is modeled with a row of four elements: CPS4R or C3D8R elements in the purely mechanical analyses; CPS4RT or C3D8RT elements in the thermomechanical analyses. The second configuration is the same as the first configuration, but one row of elements is replaced by a fixed analytical rigid surface. The bodies are initially parallel and are separated by the thickness of the interface (i.e., zero gap after accounting for the thickness). Half the nodes lie along the contact interface and are bonded.

In the purely mechanical analyses in which both bodies are modeled with elements, boundary conditions are applied to the nonbonded nodes on one of the bodies. Three separate loading conditions are applied to the other body to generate the following stress states in the interface: uniform normal stress without yielding, uniform shear stress, and nonuniform normal stress causing significant yielding at one end of the interface.

In the thermomechanical analyses in which both bodies are modeled with elements, the nonbonded nodes on both bodies are held fixed. An initial temperature of 100° is given to one body; an initial temperature of 0° is given to the other body. The temperature differential causes heat to flow between the bodies, resulting in a uniform temperature of 50° in both bodies.

In the thermomechanical analyses containing an analytical rigid surface, boundary conditions are applied to the nonbonded nodes of the deformable body to generate a uniform normal stress without yielding. The reference node of the rigid body is held fixed. An initial temperature of 100° is given to the rigid body; an initial temperature of 0° is given to the deformable body. The heat capacitance of the rigid body is defined to match that of the deformable body so that the temperature differential between the bodies will result in a uniform temperature of 50° in both bodies at the end of the analyses.

Results and discussion

Displacement results are compared to solutions obtained from the linear softening behavior models available in Abaqus. Nodal temperature results are compared to solutions obtained with the thermal conductance of the interface. The results agree for all cases.

Input files

vuinter2d_n.inp	Two-dimensional analysis that results in uniform, elastic, normal response of the interface (modeled with user subroutine *VUINTER*).
vuinter2d_n.f	User subroutine *VUINTER* used in vuinter2d_n.inp.
vuinter3d_n.inp	Three-dimensional analysis that results in uniform, elastic, normal response of the interface (modeled with user subroutine *VUINTER*).
vuinter3d_n.f	User subroutine *VUINTER* used in vuinter3d_n.inp.
linsoft2d_n.inp	Two-dimensional analysis that results in uniform, elastic, normal response of the interface (modeled with a linear softening model available in Abaqus).
linsoft3d_n.inp	Three-dimensional analysis that results in uniform, elastic, normal response of the interface (modeled with a linear softening model available in Abaqus).

vuinter2d_s.inp	Two-dimensional analysis that results in uniform, elastic, shear response of the interface (modeled with user subroutine *VUINTER*).
vuinter2d_s.f	User subroutine *VUINTER* used in vuinter2d_s.inp.
vuinter3d_s.inp	Three-dimensional analysis that results in uniform, elastic, shear response of the interface (modeled with user subroutine *VUINTER*).
vuinter3d_s.f	User subroutine *VUINTER* used in vuinter3d_s.inp.
linsoft2d_s.inp	Two-dimensional analysis that results in uniform, elastic, shear response of the interface (modeled with a linear softening model available in Abaqus).
linsoft3d_s.inp	Three-dimensional analysis that results in uniform, elastic, shear response of the interface (modeled with a linear softening model available in Abaqus).
vuinter2d_ht.inp	Two-dimensional thermomechanical analysis using user subroutine *VUINTER* to model perfect heat conduction with a constant conductance.
vuinter2d_ht.f	User subroutine *VUINTER* used in vuinter2d_ht.inp.
vuinter3d_ht.inp	Three-dimensional thermomechanical analysis using user subroutine *VUINTER* to model perfect heat conduction with a constant conductance (C3D8RT elements).
vuinter3d_ht_sc8rt.inp	Three-dimensional thermomechanical analysis using user subroutine *VUINTER* to model perfect heat conduction with a constant conductance (SC8RT elements).
vuinter3d_ht.f	User subroutine *VUINTER* used in vuinter3d_ht.inp and vuinter3d_ht_sc8rt.inp.
linsoft2d_ht.inp	Two-dimensional thermomechanical analysis with a constant gap conductance and no gap heat generation.
linsoft3d_ht.inp	Three-dimensional thermomechanical analysis with a constant gap conductance and no gap heat generation (C3D8RT elements).
linsoft3d_ht_sc8rt.inp	Three-dimensional thermomechanical analysis with a constant gap conductance and no gap heat generation (SC8RT elements).
vuinter2d_anl.inp	Two-dimensional thermomechanical analysis using user subroutine *VUINTER* with an analytical surface.
vuinter2d_anl.f	User subroutine *VUINTER* used in vuinter2d_anl.inp.
vuinter3d_anl.inp	Three-dimensional thermomechanical analysis using user subroutine *VUINTER* with an analytical surface.
vuinter3d_anl.f	User subroutine *VUINTER* used in vuinter3d_anl.inp.
linsoft2d_anl.inp	Two-dimensional thermomechanical analysis with analytical contact.
linsoft3d_anl.inp	Three-dimensional thermomechanical analysis with analytical contact (C3D8RT elements).
linsoft3d_anl_sc8rt.inp	Three-dimensional thermomechanical analysis with analytical contact (SC8RT elements).
vuinter2d_pl.inp	Two-dimensional analysis that results in plastic response of the interface (modeled with user subroutine *VUINTER*).
vuinter2d_pl.f	User subroutine *VUINTER* used in vuinter2d_pl.inp.
vuinter3d_pl.inp	Three-dimensional analysis that results in plastic response of the interface (modeled with user subroutine *VUINTER*).
vuinter3d_pl.f	User subroutine *VUINTER* used in vuinter3d_pl.inp.

VUINTERACTION

Products: Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Features tested

User subroutine to specify stress and heat flux between contacting surfaces when using general contact.

Problem description

User subroutine *VUINTERACTION* used in this example models a mechanically compliant, thermally conductive interface material with uniform thickness. The interface material is assumed to be bonded to each of two contacting surfaces. The interface material exhibits elastic-plastic behavior with linear hardening in the normal contact direction and purely elastic resistance to relative sliding. Membrane straining of the interface does not affect the stress transmitted to the surfaces. The interface material is thermally conductive; and the conductance remains constant, independent of the gap or pressure at the interface.

Two simple configurations are used to test this user subroutine. In the first configuration each of two identical elastic bodies is modeled with a row of four elements. The second configuration is the same as the first configuration, but one row of elements is replaced by an analytical rigid surface. The bodies are initially parallel and bonded along the contact interface.

A complete list of properties specified for this interface model, in the order in which the values are specified on the second data line of the contact property definition, is as follows:

1. Gap cutoff distance. Only secondary nodes with initial gaps less than this distance are bonded.
2. Young's modulus of the interface material.
3. Poisson's ratio of the interface material.
4. Initial yield stress in the normal direction of the interface material.
5. Hardening modulus in the normal direction of the interface material.
6. Interface thickness used in the strain calculations.
7. Thermal conductivity (units of $\mathbf{J}\boldsymbol{\theta}^{-1}\mathbf{T}^{-1}\mathbf{L}^{-1}$) of the interface material used for contact heat flux calculations.

Load cases

In the purely mechanical analyses the interaction of the two elastic bodies is introduced through boundary conditions on the nodes away from the interface. Three separate loading conditions are applied to generate the following stress states in the interface: uniform normal stress without yielding, uniform shear stress, and nonuniform normal stress causing significant yielding at one end of the interface. For the first two cases the solution is compared with that of a reference model that uses a linear softening interface behavior available in Abaqus. For the last case with plasticity, the solution is compared with that of a reference model that uses subroutine *VUINTER* for defining the interface response. In addition, for the last case a gap is introduced between the two elastic bodies to account for the thickness of the interface material.

In the thermomechanical analyses the two elastic bodies are held fixed with a gap between them. An initial temperature of 100° is given to one body; an initial temperature of 0° is given to the other body. The temperature differential causes heat flow between the bodies, resulting in a steady-state temperature of 50° in both bodies. The solution is compared to that obtained with the interface thermal conductance.

Tracking thickness and other issues specific to VUINTERACTION functionality

The two surfaces, identified for interaction, are tracked to identify those segments of these surfaces that are within a fixed distance, called tracking thickness, and those segments are made available in the user subroutine *VUINTERACTION* for defining their interaction. Hence, the tracking thickness must be set greater or equal to the maximum anticipated interface material thickness.

The state variables are associated with secondary nodes and can be passed to user subroutine *VUINTERACTION* multiple times within an increment as a given secondary node may be within tracking distance to multiple main facets. In the elastic-plastic analysis using *VUINTERACTION*, two state variables are used to keep track of the current yield stress. During the solution the previous yield stress is read from state variable 1 and the new yield stress is written to state variable 2 for time increments that are odd; the previous yield stress is read from state variable 2, and the new yield stress is written to state variable 1 for time increments that are even. This setup is incorporated to avoid using a state variable that has already been updated in the current time increment.

When thermal interaction is active between the surfaces, the heat fluxes are calculated by multiplying the thermal conductivity of the interface material with the temperature difference between the secondary node and main contact point and dividing by the initial interface thickness. Effects such as heat generation due to friction are not taken into account.

Results and discussion

Displacement results in the pure mechanical interaction models and the nodal temperature results in the themo-mechanical interaction models are compared to their respective reference solutions. The results agree for all cases.

Input files

Mechanical interaction between surfaces

vuinteraction_normal.inp	Loading normal to the elastic interface.
vuinteraction_normal.f	User subroutine *VUINTERACTION* used with vuinteraction_normal.inp.
dfltpp3d_normal.inp	Reference model for comparison with results of vuinteraction_normal.inp.
vuinteraction_shear.inp	Loading tangential to the elastic interface.
vuinteraction_shear.f	User subroutine *VUINTERACTION* used with vuinteraction_shear.inp.
dfltpp3d_shear.inp	Reference model for comparison with results of vuinteraction_shear.inp.
vuinteraction_rbody_normal.inp	Same as vuinteraction_normal.inp except one of the two elastic bodies is made rigid.
vuinteraction_rbody_normal.f	User subroutine *VUINTERACTION* used with vuinteraction_rbody_normal.inp.
dfltpp3d_rbody_normal.inp	Reference model for comparison with results of vuinteraction_rbody_normal.inp.
vuinteraction_plastic.inp	Loading normal to the elastic plastic interface.

vuinteraction_plastic.f	*VUINTERACTION* used with vuinteraction_plastic.inp.
vuinter_plastic.inp	Reference model for comparison with results of vuinteraction_plastic.inp.
vuinter_plastic.f	User subroutine *VUINTER* used with vuinter_plastic.inp.

Thermal interaction between surfaces

vuinteraction_heat.inp	Heat transfer across the interface.
vuinteraction_heat.f	User subroutine *VUINTERACTION* used with vuinteraction_heat.inp.
dfltpp3d_heat.inp	Reference model for comparison with results of vuinteraction_heat.inp.

VUMAT

Products: Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

CPE4R C3D8R M3D4R S4R

Features tested

Large deformation kinematics, elastic-plastic material with strain hardening, user material, multi-point constraints.

Problem description

The rotating cylinder problem was proposed by *Longcope and Key (1977)* as a means of exercising finite rotation algorithms. In this problem a cylinder with an initial angular velocity of 4000 rad/sec and a zero initial stress state is modeled. (This is physically impossible because the body forces would generate a stress field under this angular velocity. Nevertheless, these initial conditions are acceptable, since this is merely a numerical experiment.) The inside of the cylinder is subjected to an instantaneous application of a pressure of 67.3 MPa (9760 psi).

The elastic material properties are defined as Young's modulus of 71 GPa (1.03×10^7 psi), Poisson's ratio of 0.3333, and density of 2680 kg/m^3 (2.508×10^{-4} lb sec^2 in^{-4}). An isotropic hardening plasticity model is used with an initial yield of 286 MPa (4.15×10^4 psi) and constant hardening modulus of 3.565 GPa (5.17×10^5 psi).

Only one-quarter of the ring is modeled using a constraint equation and a multi-point constraint to enforce the repeated symmetry boundary condition.

A local cylindrical coordinate system is defined at each material point of the mesh.

Results and discussion

The first case considered is a two-dimensional model using CPE4R elements. In this case two meshes are defined in the same problem, as shown in *Figure 1*. The lower mesh in *Figure 1* uses the built-in Mises isotropic hardening plasticity model. The upper mesh in *Figure 1* employs user subroutine *VUMAT* with the kinematic hardening Mises model described in *Models for Metals Subjected to Cyclic Loading*. *Figure 2* shows the time history of the maximum principal stress in the two-dimensional model for both cases. *Figure 3* shows the time history of equivalent plastic strain in the two-dimensional model for both cases. *Figure 4* shows the energy histories in the two-dimensional model. The energy history is particularly important in this analysis because it demonstrates that there is no energy lost in the enforcement of multi-point constraints.

The second case is a three-dimensional representation of the same problem using shells, membranes, and brick elements to model the ring with suitable boundary conditions to reproduce closely the original two-dimensional model. The built-in Mises isotropic hardening plasticity model is used. The meshes for the three-dimensional case are shown in *Figure 5*. *Figure 6* shows the time history of the maximum principal stress in the three-dimensional model for both cases. *Figure 7* shows the time history of the equivalent plastic strain in the three-dimensional model for both cases. *Figure 8* shows the energy histories in the three-dimensional model. Note that each energy quantity is summed over the two cases.

The results compare well with those obtained by *Longcope and Key (1977)*.

Input files

rotcyl2d.inp	Input data for the two-dimensional case.
rotcyl2dvumat.f	*VUMAT* subroutine for the two-dimensional case.
rotcyl2d_effMod.inp	Input data for the two-dimensional case with effective modulus defined in user subroutine *VUMAT*.
rotcyl2dvumat_effMod.f	*VUMAT* subroutine for the two-dimensional case.
rotcyl3d.inp	Input data for the three-dimensional case.

References

- Longcope, D. B., and S. W. Key, "On the Verification of Large Deformation Inelastic Dynamic Calculations through Experimental Comparisons and Analytic Solutions," PVP-PB-023, American Society of Mechanical Engineers, 1977.

Figures

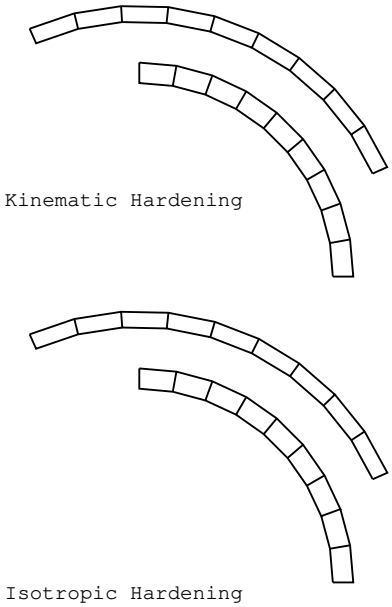


Figure 1: Mesh for the two-dimensional case.

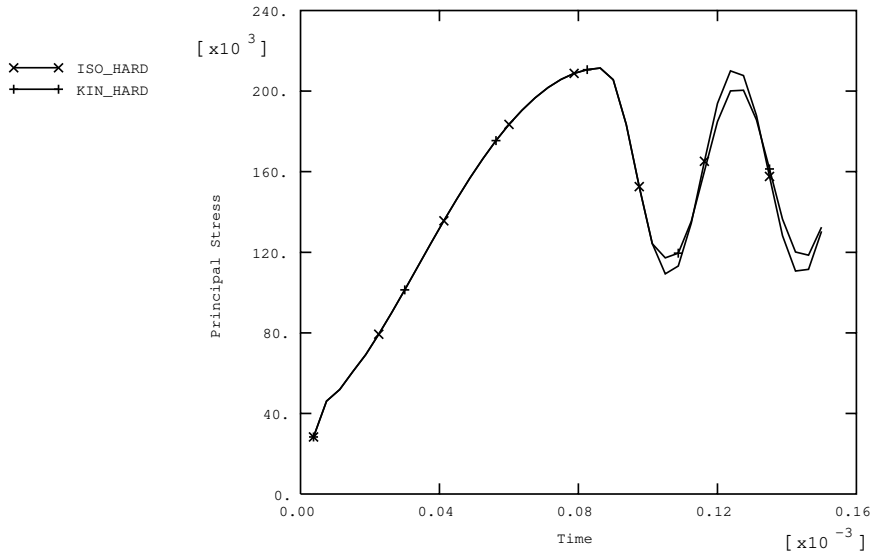


Figure 2: Maximum principal stress versus time for the two-dimensional case.

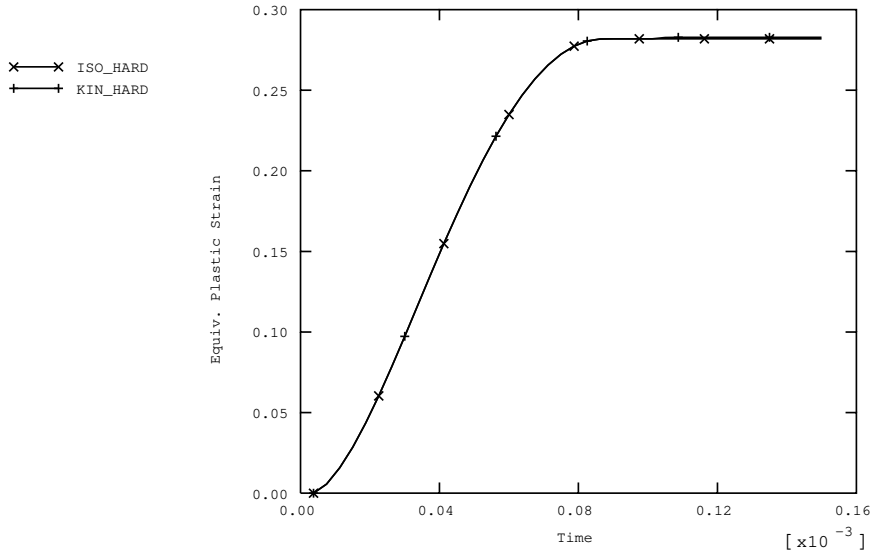


Figure 3: Equivalent plastic strain versus time for the two-dimensional case.

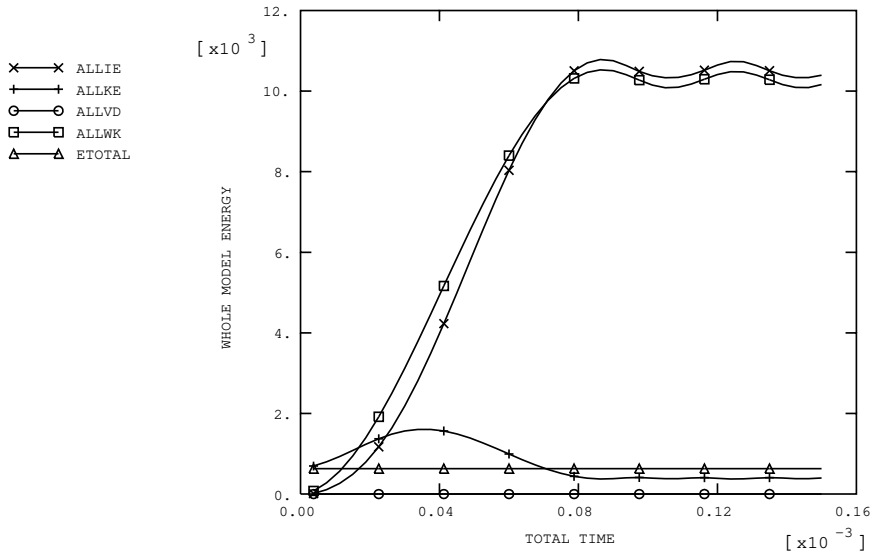


Figure 4: Energy histories for the two-dimensional case.

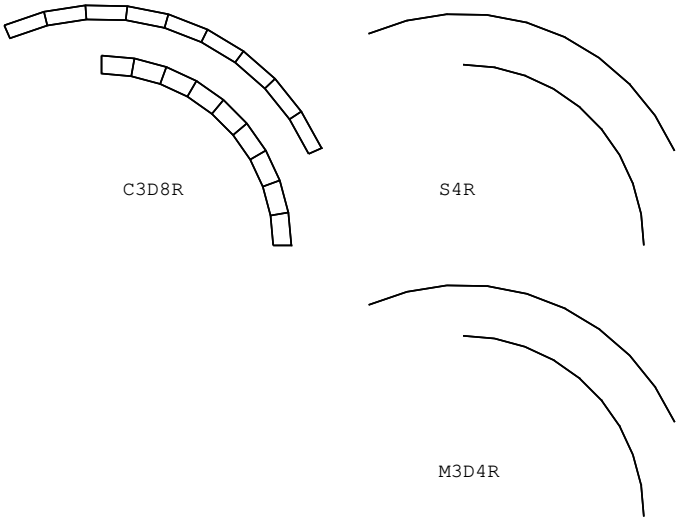


Figure 5: Meshes for the three-dimensional case.

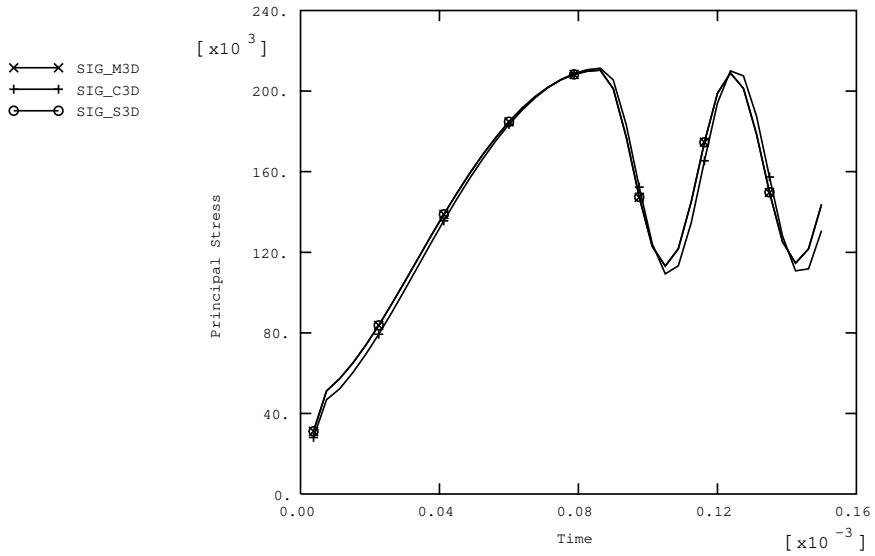


Figure 6: Maximum principal stress versus time for the three-dimensional case.

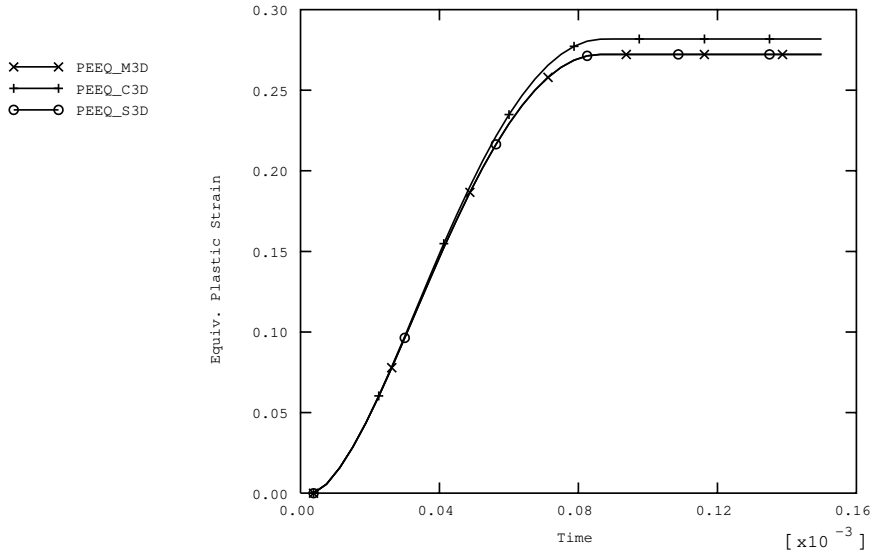


Figure 7: Equivalent plastic strain versus time for the three-dimensional case.

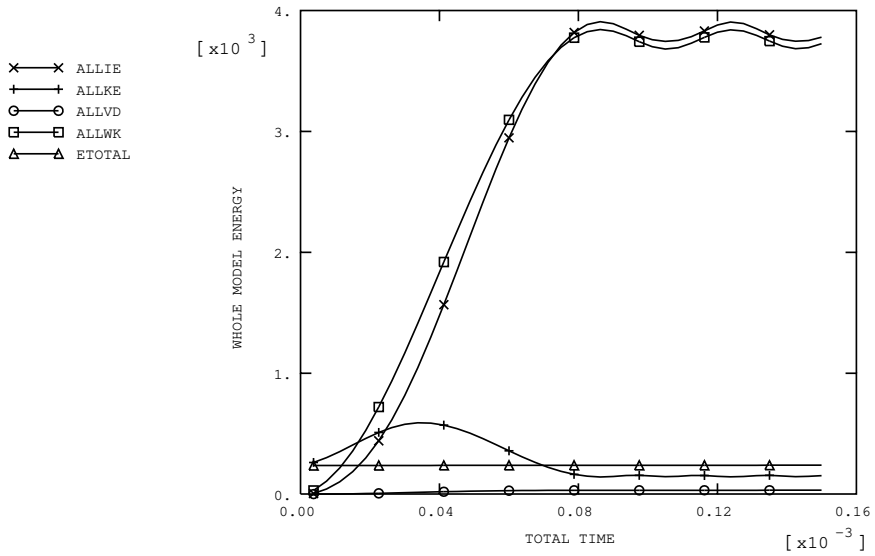


Figure 8: Energy histories for the three-dimensional case.

VUMATHT

Products: Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

CPS3T CPS4RT

Features tested

User subroutine to define a simple thermal constitutive behavior.

Problem description

User subroutine *VUMATHT* is used to define the thermal behavior of the material for a coupled temperature-displacement analysis in Abaqus/Explicit. Isotropic conductivity and a constant specific heat for the material are assumed. The heat conduction in the material is governed by Fourier's law, and the gradient of the heat flux with respect to temperature is zero. The material properties—namely, conductivity and specific heat—are defined and are passed into the user subroutine via the `PROPS` array. Dummy mechanical properties are prescribed to complete the material definition.

Verification problems *tempdepfilm_xpl_cps3t.inp* and *tempdepfilm_xpl_cps4rt.inp* solve the same problem using the standard data line options to define conductivity and specific heat (see *Temperature-dependent film condition*).

Results and discussion

The results of *tempdepfilm_xpl_cps3t_vumatht.inp* match those obtained from *tempdepfilm_xpl_cps3t.inp*.

The results of *tempdepfilm_xpl_cps4rt_vumatht.inp* match those obtained from *tempdepfilm_xpl_cps4rt.inp*.

Input files

tempdepfilm_xpl_cps3t_vumatht.inp	Analysis using CPS3T elements; the thermal material behavior is defined using user subroutine *VUMATHT* in vumathtxx.f.
tempdepfilm_xpl_cps4rt_vumatht.inp	Analysis using CPS4RT elements; the thermal material behavior is defined using user subroutine *VUMATHT* in vumathtxx.f.
vumathtxx.f	User subroutine *VUMATHT* defining a simple thermal constitutive behavior.

VUSDFLD

Products: Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

B31 C3D8 C3D8R CAX4R CPE4 CPE4R CPS4R
M3D4 M3D4R PIPE31 S4 S4R S4RS SAX1 SC8R T2D2 T3D2

Features tested

User subroutine to redefine field variables at a material point.

Problem description

This test verifies that the user-defined material point field variable in user subroutine *VUSDFLD* is transferred properly to the solution process. The finite element model consists of multiple disconnected elements of the types listed above. Each element is associated with a Mises plasticity model and in each case a damage model is constructed based on a user-defined field. For comparison purposes, a duplicate set of elements with equivalent damage initiation/damage evolution definitions is included to provide a reference solution.

Results and discussion

In each case the results in the testing element match the solution in the reference element, which can be observed from the time-history plots of the output variables.

Input files

damage_vusdfld.inp	Damage analysis.
damage_vusdfld.f	User subroutine *VUSDFLD* used in damage_vusdfld.inp.

VUTRSNETWORK

Products: Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D8

Features tested

User subroutine to define a reduced time shift function for models defined within the parallel rheological framework.

Problem description

The input file consists of one element with the material response specified using the parallel rheological framework with two viscous networks. The reduced times were used with the shift function approximated by the Williams-Landel-Ferry form in network one and by the Arrhenius form in network two, both defined in user subroutine *VUTRSNETWORK*. A load is applied quickly to the element in the first step and is held constant throughout the second step so that the material is subjected to creep. The temperature is also increased during the second step. The results are compared to the results obtained with an almost identical model that uses the same shift functions specified using the built-in Abaqus functionality.

Results and discussion

The results obtained using the user subroutine and the built-in Abaqus models are identical.

Input files

x_viscnet_c3d8_nh_n2_trs_user.inp	The model that uses user subroutine *VUTRSNETWORK* to define the Williams-Landel-Ferry and Arrhenius shift functions.
x_viscnet_wlf_arrhenius_trs.f	User subroutine to define the Williams-Landel-Ferry and Arrhenius forms of the shift function.

VUVISCOSITY

Products: Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D8R CPE4R

Features tested

User subroutine to define the viscosity for equation of state models with viscous shear behavior.

Problem description

This test verifies that the user-defined viscosity in user subroutine *VUVISCOSITY* is transferred properly to the solution process. The finite element model consists of a C3D8R element and a CPE4R element with identical material properties. The viscous properties of the material are defined with a Cross viscosity model that is implemented as a user-defined viscosity. For comparison purposes, a duplicate set of elements with equivalent viscosity definitions is included to provide a reference solution.

Results and discussion

In each case the results in the testing element match the solution in the reference element.

Input files

cross_vuviscosity.inp	Cross viscosity model, shear test.
cross_vuviscosity.f	User subroutine *VUVISCOSITY* used in cross_vuviscosity.inp.

VWAVE

Products: Abaqus/Explicit Abaqus/Aqua

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Features tested

User subroutine *VWAVE* is used to specify wave kinematics for an Abaqus/Aqua analysis through Abaqus/Explicit.

PB_VWAVE-AIRY-PERIOD_P31_XPL.inp

Elements tested

B31 PIPE31

Problem description

Verification problem *pb_vwave-airy-period_p31_xpl.inp* is identical to *pb_airy_p31_xpl.inp* as described in *Dynamic pressure, closed-end buoyancy loads*. It tests the dynamic pressure implementation and closed-end buoyancy loading for Airy waves coded in user subroutine *VWAVE*. In this problem a vertical pile is fully constrained and subjected to buoyancy loading. Airy wave theory is used to calculate the total reaction force on the structure during a dynamic analysis. Distributed load type PB is used with a 50-element (PIPE31) model, and concentrated load type TSB is used with a one-element (B31) model.

Model:

Height of the structure	175.0 (100.0 below and 75.0 above mean water elevation)
Pipe section data	$r = 1.0$, $t = 0.25$

Material:

Young's modulus	1×10^6

Aqua—environment:

Seabed elevation	100.0
Mean water elevation	1100.0
Gravitational constant	32.2
Fluid mass density	2.0

Results and discussion

The results are identical to those calculated in *pb_airy_p31_xpl.inp* and agree well with the analytically determined peak total reaction force.

Input files

pb_vwave-airy-period_p31_xpl.inp	Input file for this analysis.
pb_vwave-airy-length_p31_xpl.inp	Same analysis, but the corresponding wave length is specified instead of the wave period.
vwave_airy.f	User subroutine *VWAVE* used for pb_vwave-airy-period_p31_xpl.inp and pb_vwave-airy-length_p31_xpl.inp.
pb_vwave-airy-period-opt_p31_xpl.inp	The wave length calculation from the wave period might be expensive. This optimized input file calculates the wave length once initially and stores the wave parameters appropriately for subsequent usage.
vwave_airy-opt.f	User subroutine *VWAVE* used for pb_vwave-airy-period-opt_p31_xpl.inp.

PB_VWAVE-AIRY-PERIOD_P21_XPL.inp

Elements tested

B21 PIPE21

Problem description

Verification problem *pb_vwave-airy-period_p21_xpl.inp* is identical to *pb_airy_p21_xpl.inp* as described in *Dynamic pressure, closed-end buoyancy loads*. It tests the dynamic pressure implementation and closed-end buoyancy loading for Airy waves coded in user subroutine *VWAVE*. In this problem a vertical pile is fully constrained and subjected to buoyancy loading. Airy wave theory is used to calculate the total reaction force on the structure during a dynamic analysis. Distributed load type PB is used with a 50-element (PIPE21) model, and concentrated load type TSB is used with a one-element (B21) model.

Model:

Height of the structure	175.0 (100.0 below and 75.0 above mean water elevation)
Pipe section data	$r = 1.0$, $t = 0.25$

Material:

Young's modulus	1×10^6

Aqua—environment:

Seabed elevation	100.0
Mean water elevation	1100.0
Gravitational constant	32.2
Fluid mass density	2.0

Results and discussion

The results are identical to those calculated in *pb_airy_p21_xpl.inp* and agree well with the analytically determined peak total reaction force.

Input files

pb_vwave-airy-period_p21_xpl.inp	Input file for this analysis.
pb_vwave-airy-length_p21_xpl.inp	Same analysis, but the corresponding wave length is specified instead of the wave period.
vwave_airy.f	User subroutine *VWAVE* used for pb_vwave-airy-period_p21_xpl.inp and pb_vwave-airy-length_p21_xpl.inp.
pb_vwave-airy-period-opt_p21_xpl.inp	The wave length calculation from the wave period might be expensive. This optimized input file calculates the wave length once initially and stores the wave parameters appropriately for subsequent usage.
vwave_airy-opt.f	User subroutine *VWAVE* used for pb_vwave-airy-period-opt_p21_xpl.inp.

Miscellaneous Modeling Options

In this section:

- *Activation functions used by feed-forward neural networks*
- *Adaptive mesh for solid elements in Abaqus/Standard*
- *Adjusting nodal coordinates*
- *Amplitude*
- *Spatially varying element properties*
- *Define a part instance that will be used for display only*
- *Establishing geostatic equilibrium when the initial stress state is unknown*
- *Specifying geometric imperfection and parameter shape variation*
- *Specifying user-defined field variables directly in the input file*
- *Inertia-based load balancing*
- *Defining the cutting surface*
- *Defining a matrix for part of a model*
- *User-defined coordinate system*
- *Pre-tension section*
- *Radiation view factor definition: symmetries and blocking*
- *Release rotational degrees of freedom*
- *Extrapolation*
- *Surface-based fluid cavities*
- *Defining temperature, field variable, pressure stress, and pore fluid pressure values*
- *Coupled pore-thermal elements*

Activation functions used by feed-forward neural networks

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

T2D3

Features tested

Activation functions used by feed-forward neural networks using multiple neural networks defined in the same input file.

Problem description

Abaqus/Standard uses a simple two-dimensional wire model meshed with truss elements to test the activation functions supported by feed-forward neural networks.

The model contains multiple feed-forward neural networks. All neural networks are defined with both the input and output vectors having a dimension of one. Each neural network uses a single neuron with a different activation function. The neuron has a single weight with a unit value and the bias set to zero.

The analysis is performed over multiple static steps, with each step using a different neural network to prescribe the U2 displacements at all nodes in the model by evaluating the neural network from user subroutine *DISP*.

Results and discussion

Either the Fortran or the C/C++ version of user subroutine *DISP* can be used to apply the displacements during the analysis. Inside *DISP*, the step number is used as the ID of the neural network to be evaluated for calculating the displacement in the vertical direction at each node based on its x-coordinate in the initial configuration.

Input files

ffd_nn_act_funcs_truss.inp	Model using multiple **NEURAL NETWORK* definitions, evaluated from user subroutine *DISP* to apply displacements during each step.
ffd_nn_act_funcs_truss_disp.f	Fortran version of user subroutine *DISP* used with input file `ffd_nn_act_funcs_truss.inp`.
ffd_nn_act_funcs_truss_dispC.c	C/C++ version of user subroutine *DISP* used with input file `ffd_nn_act_funcs_truss.inp`.

Figures

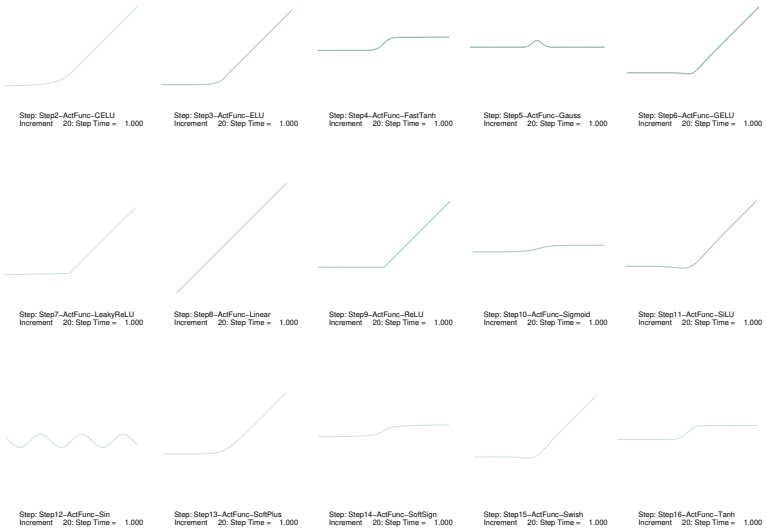


Figure 1: Deformed wire shapes at the end of each step corresponding to the activation function used during the step.

Adaptive mesh for solid elements in Abaqus/Standard

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

CPE3 CPE3H CPS3 CAX3 CAX3H CPS4 CPS4T CPE4 CPE4T CPE4H CPE4HT
CPE4P CPE4PH CAX4 CAX4P CAX4H CAX4T CAX4HT CAX4PH
C3D8 C3D8P C3D8H C3D8T C3D8HT C3D8PH
C3D8R C3D8RP C3D8RH C3D8RT C3D8RHT C3D8RPH

Features tested

Adaptive mesh capabilities are tested in Abaqus/Standard for solid elements that can be part of an adaptive mesh domain.

Problem description

The verification problems are either slender beam-like structures that are loaded by gravity parallel to the length or cubical structures indented by a rigid punch.

The verification problems also test user subroutine *UMESHMOTION*, which provides user-prescribed mesh motion.

Results and discussion

The verification of the adaptive mesh capability is done by comparing the results of the problems with and without adaptive mesh options.

The verification of user subroutine *UMESHMOTION* consists of checking the nodal output to ensure correct application of the user-prescribed mesh motion.

Input files

ale_foamindent_cpe3.inp	Punch indentation problem using CPE3 elements.
ale_foamindent_cpe3_gv.inp	Punch indentation problem using CPE3 elements, and volume-based smoothing with geometric enhancements.
ale_foamindent_cpe3h.inp	Punch indentation problem using CPE3H elements.
ale_foamindent_cpe3h_gv.inp	Punch indentation problem using CPE3H elements, and volume-based smoothing with geometric enhancements.
ale_foamindent_cps3.inp	Punch indentation problem using CPS3 elements.
ale_foamindent_cps3_gv.inp	Punch indentation problem using CPS3 elements, and volume-based smoothing with geometric enhancements.
ale_foamindent_cax3.inp	Punch indentation problem using CAX3 elements.

ale_foamindent_cax3_surf.inp	Punch indentation problem with CAX3 elements using surface-to-surface contact.
ale_foamindent_cax3_gv.inp	Punch indentation problem using CAX3 elements, and volume-based smoothing with geometric enhancements.
ale_foamindent_cax3h.inp	Punch indentation problem using CAX3H elements.
ale_foamindent_cax3h_gv.inp	Punch indentation problem using CAX3H elements, and volume-based smoothing with geometric enhancements.
ale_foamindent_cps4.inp	Punch indentation problem using CPS4 elements.
ale_foamindent_cps4_gv.inp	Punch indentation problem using CPS4 elements, and volume-based smoothing with geometric enhancements.
ale_foamindent_cpe4.inp	Punch indentation problem using CPE4 elements.
ale_foamindent_cpe4_gv.inp	Punch indentation problem using CPE4 elements, and volume-based smoothing with geometric enhancements.
ale_foamindent_cpe4h.inp	Punch indentation problem using CPE4H elements.
ale_foamindent_cpe4h_gv.inp	Punch indentation problem using CPE4H elements, and volume-based smoothing with geometric enhancements.
ale_foamindent_cax4.inp	Punch indentation problem using CAX4 elements.
ale_foamindent_cax4h_gv.inp	Punch indentation problem using CAX4H elements, and volume-based smoothing with geometric enhancements.
ale_cps4t.inp	Cantilever under gravity loading using CPS4T elements.
ale_cpe4t.inp	Cantilever under gravity loading using CPE4T elements.
ale_cpe4ht.inp	Cantilever under gravity loading using CPE4HT elements.
ale_cpe4p.inp	Cantilever under gravity loading using CPE4P elements.
ale_cpe4ph.inp	Cantilever under gravity loading using CPE4PH elements.
ale_cax4t.inp	Cantilever under gravity loading using CAX4T elements.
ale_cax4ht.inp	Cantilever under gravity loading using CAX4HT elements.
ale_cax4p.inp	Cantilever under gravity loading using CAX4P elements.
ale_cax4ph.inp	Cantilever under gravity loading using CAX4PH elements.
ale_c3d8.inp	Cantilever under gravity loading using C3D8 elements.
ale_c3d8_gv.inp	Cantilever under gravity loading using C3D8 elements, and volume-based smoothing with geometric enhancements.
ale_c3d8p.inp	Cantilever under gravity loading using C3D8P elements.
ale_c3d8h.inp	Cantilever under gravity loading using C3D8H elements.
ale_c3d8h_gv.inp	Cantilever under gravity loading using C3D8H elements, and volume-based smoothing with geometric enhancements.
ale_c3d8t.inp	Cantilever under gravity loading using C3D8T elements.
ale_c3d8ht.inp	Cantilever under gravity loading using C3D8HT elements.
ale_c3d8ph.inp	Cantilever under gravity loading using C3D8PH elements.
ale_c3d8r.inp	Cantilever under gravity loading using C3D8R elements.
ale_c3d8r_gv.inp	Cantilever under gravity loading using C3D8R elements, and volume-based smoothing with geometric enhancements.
ale_c3d8rp.inp	Cantilever under gravity loading using C3D8RP elements.
ale_c3d8rh.inp	Cantilever under gravity loading using C3D8RH elements.
ale_c3d8rh_gv.inp	Cantilever under gravity loading using C3D8RH elements, and volume-based smoothing with geometric enhancements.

ale_c3d8rt.inp	Cantilever under gravity loading using C3D8RT elements.
ale_c3d8rht.inp	Cantilever under gravity loading using C3D8RHT elements.
ale_c3d8rph.inp	Cantilever under gravity loading using C3D8RPH elements.
ale_c3d8_cavityablation.inp	Cavity ablation using user subroutine *UMESHMOTION*.
ale_c3d8_cavityablation_gv.inp	Cavity ablation using user subroutine *UMESHMOTION*, and volume-based smoothing with geometric enhancements.
ale_c3d8_cavityablation.f	User subroutine *UMESHMOTION* used in ale_c3d8_cavityablation.inp and ale_c3d8_cavityablation_gv.inp.
ale_c3d8_uniformablation.inp	Uniform ablation using user subroutine *UMESHMOTION*.
ale_c3d8_uniformablation_gv.inp	Uniform ablation using user subroutine *UMESHMOTION*, and volume-based smoothing with geometric enhancements.
ale_c3d8_uniformablation.f	User subroutine *UMESHMOTION* used in ale_c3d8_uniformablation.inp and ale_c3d8_uniformablation_gv.inp.
ale_cpe4p_cavityablation.inp	Cavity ablation using user subroutine *UMESHMOTION*.
ale_cpe4p_cavityablation.f	User subroutine *UMESHMOTION* used in ale_cpe4p_cavityablation.inp.
ale_cpe4p_uniformablation.inp	Uniform ablation using user subroutine *UMESHMOTION*.
ale_cpe4p_uniformablation.f	User subroutine *UMESHMOTION* used in ale_cpe4p_uniformablation.inp.
ale_constraint.inp	Verification to ensure proper boundary condition application in an adaptive mesh domain.
ale_constraint_gv.inp	Verification to ensure proper boundary condition application in an adaptive mesh domain. Volume-based smoothing and geometric enhancements are applied.

Adjusting nodal coordinates

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

AC2D3 AC2D4 AC2D4R AC2D6 AC2D8

Features tested

The adjustment of nodal coordinates is tested.

Problem description

The test verifies the adjustment of nodal coordinates. The reference nodes of three coupling definitions are adjusted to their corresponding surfaces.

Results and discussion

The adjust nodal coordinates are printed to the data (`.dat`) file. These new nodal coordinates verify that nodes have been adjusted to their corresponding surfaces.

Input files

fast_adjust.inp	Adjust three different coupling reference nodes to their corresponding surfaces.

Amplitude

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

CPE4R MASS T3D2

Features tested

Several methods of specifying time variations of prescribed magnitudes are tested through the use of the amplitude curve.

Problem description

The amplitude curve is used to specify a function that defines arbitrary time variations of prescribed variables throughout an analysis. The user can specify this function with a variety of methods. Two of the methods use tabulated values that define a continuous function of linear segments. The tabular amplitude definition uses a nonfixed time increment, which requires that pairs of time-amplitude data be supplied. The equally spaced amplitude definition uses a fixed time increment that is specified once, and only the values of the function are required. Two other amplitude types use trigonometric functions to define the function. The periodic amplitude definition uses the Fourier series to define the function. The modulated amplitude definition uses the product of two sine functions. The exponential decay amplitude definition uses an exponential function. The smooth-step amplitude definition uses a fifth-order polynomial equation to ramp up/down smoothly from one amplitude value to the next. A solution-dependent amplitude definition (available only in Abaqus/Standard) accepts a starting value and lets Abaqus calculate subsequent values based on the evolution of solution parameters. Currently there is only one solution parameter available, the maximum equivalent creep strain rate, which is compared to target values entered in the creep strain rate.

If the function describes either a displacement or velocity in a dynamic analysis, the derivatives and integrations of the function are required. For the three amplitude types that use trigonometric or exponential functions, the derivatives are continuous and available. For the amplitude type that uses a fifth-order polynomial equation, the derivatives are continuous and available; however, both the first and second derivatives are zero at the data point. For the two types that use tabulated values, the linear segments do not have continuous derivatives, and the second derivative will be infinite at the segment intersections. An amplitude curve with smoothing allows the user to define an interval about the data points in which a quadratic function is interpolated to give a continuous first derivative and a finite second derivative. The use of this parameter is verified within these tests.

Input files *xampmult.inp* (Abaqus/Standard) and *xamptest.inp* (Abaqus/Explicit) are analyses performed over multiple steps during which several loads and displacements are applied in terms of defined amplitudes with various settings of the parameters. *xampresm.inp* (Abaqus/Standard) and *xamprest.inp* (Abaqus/Explicit) restart the analyses from the previous step. A simple truss model is used. Various nodal degrees of freedom are prescribed with the boundary conditions, and loads are applied. In all of these cases the prescribed quantities are defined using the amplitude curve. The purpose of this test is to ensure that the initial value of the function to be applied in the next step is interpolated properly from the amplitude definitions. Since *xampresm.inp* and *xamprest.inp* restart the analysis from the previous step, the results will show that the initial value at the beginning of the restart step is obtained from the point on the amplitude curve at which the restart was done; the value will be

ramped to the new value defined in the new step. The output variables corresponding to the prescribed input are checked to verify the use of the amplitude curve.

xampsdep.inp and *xampress.inp* simulate the superplastic forming of a rectangular pan in Abaqus/Standard. The pressure applied to a sheet that forces it to acquire the shape of a die is determined by solution-dependent amplitude.

Results and discussion

The results for each of the amplitude types are discussed in the following sections.

Tabular amplitude definition

A typical variation of a boundary condition is shown in the history plots of *Figure 1* through *Figure 3*. For this particular example of the variation, the velocity history is specified. The tabulated input is given to represent a sine curve. The acceleration history shown is the time derivative of the velocity curve. The displacement history is the integration of the velocity curve. Various other types of boundary conditions and specified curves are verified in the test.

Periodic amplitude definition

One of the boundary condition variations used in *xampmult.inp* and *xamptest.inp* is specified with the velocity history. The variation is specified using a sinusoidal variation corresponding to the following expression:

$f(x) = .5\ \sin(10\pi t)$. This variation was chosen such that it is identical to the function specified using tabulated values in the previous section. The acceleration, velocity, and displacement histories are the same as those from the previous section, as shown in *Figure 1* through *Figure 3*.

Equally spaced amplitude definition

The velocity history is specified in *xampmult.inp* and *xamptest.inp*. This particular variation is specified using a fixed time step. The variation was chosen such that it is identical to the function specified using tabulated values without fixed time steps (see *Tabular amplitude definition*"). The acceleration, velocity, and displacement histories are the same as those discussed previously and are shown in *Figure 1* through *Figure 3*.

Modulated amplitude definition

One of the boundary condition variations used in *xampmult.inp* and *xamptest.inp* is specified with the velocity history using a scale factor of 100.0. The variation is specified using a combination of sinusoidal functions corresponding to the following expression:

$$f(x) = .5 \sin[10\pi (t - .018)] \sin[20\pi (t - .018)].$$

A scale factor of 200.0 was used to magnify the function. The acceleration, velocity, and displacement histories for this particular variation are shown in the history plots of *Figure 4* through *Figure 6*.

Exponential decay amplitude definition

One of the boundary condition variations used in *xampmult.inp* and *xamptest.inp* is specified with the velocity history using a scale factor of 200. The variation is specified using an exponential function corresponding to the following expression:

$$f(x) = 4.094 \exp\left[(t - .2)/1.\right].$$

The acceleration, velocity, and displacement histories for this particular variation are shown in the history plots of *Figure 7* through *Figure 9*.

Smooth-step amplitude

One of the boundary condition variations used in *xampmult.inp* and *xamptest.inp* is specified with the velocity history using a scale factor of 100.0. The variation is specified using a polynomial equation corresponding to the following expression:

$$f(x) = 0.1 + 0.9\xi^3\left(10 - 15\xi + 6\xi^2\right),$$

where

$$\xi = (t - 0.12)/0.78.$$

The acceleration, velocity, and displacement histories for this particular variation are shown in the history plots of *Figure 10* through *Figure 12*.

Solution-dependent amplitude

The initial value of the pressure is 1.0, and the amplitude is allowed to increase to 100 times that value (as well as decrease to 0.1 times the initial value). The maximum amplitude was reached, and Abaqus/Standard stopped the analysis because it could not follow the objective within the restrictions imposed. This happened before the sheet completely filled the die cavity. A restart run, *xampress.inp*, in which the maximum amplitude is modified to 500 times the reference load allows the deformation to be completed. Once again, the maximum allowable amplitude is used as the mechanism for Abaqus/Standard to end the analysis. The restart run exemplifies another possibility that is generally not recommended (since it will probably not occur in practice)—the loading reference value was increased by a factor of 5.0. As a result, the amplitude history adapted itself accordingly.

Figure 13 and *Figure 16* show the rigid surface and the deformable sheet at different stages of deformation. *Figure 14* and *Figure 17* show the amplitude history obtained. *Figure 15* shows the ratio between the maximum creep strain rate in the model and the target value provided.

Input files

Abaqus/Standard analyses

xampmult.inp	**AMPLITUDE* used over multiple steps.
xampresm.inp	**RESTART* test of xampmult.inp.
xampsdep.inp	**AMPLITUDE*,DEFINITION=SOLUTION DEPENDENT.
xampress.inp	**RESTART* test of xampsdep.inp.

Abaqus/Explicit analyses

xamptest.inp	**AMPLITUDE* used over multiple steps.
xamprest.inp	**RESTART* test of xamptest.inp.

Figures

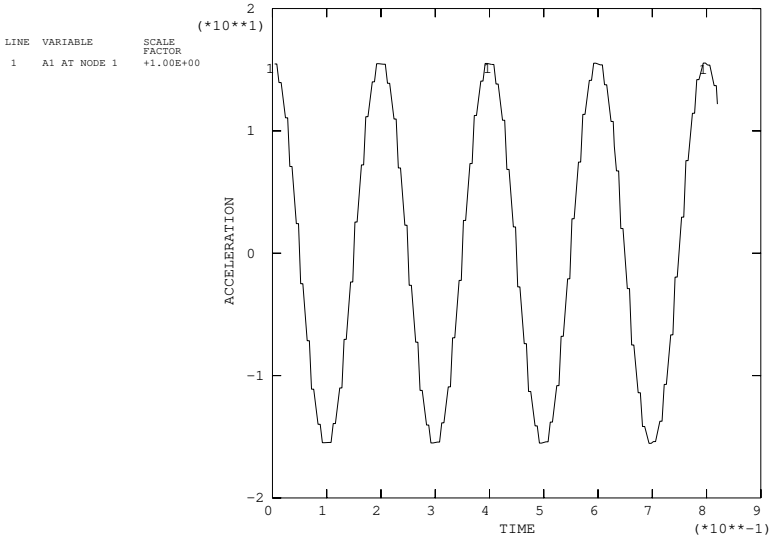


Figure 1: Acceleration history; tabular, periodic, or equally spaced amplitude definition.

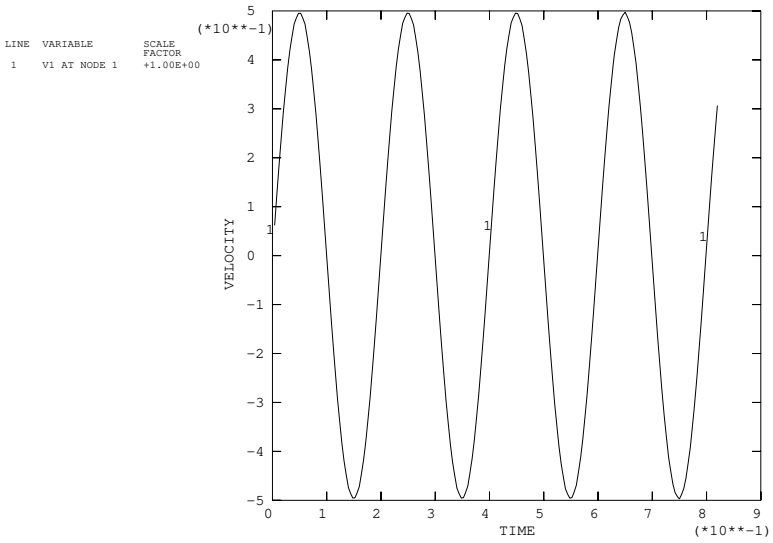


Figure 2: Velocity history; tabular, periodic, or equally spaced amplitude definition.

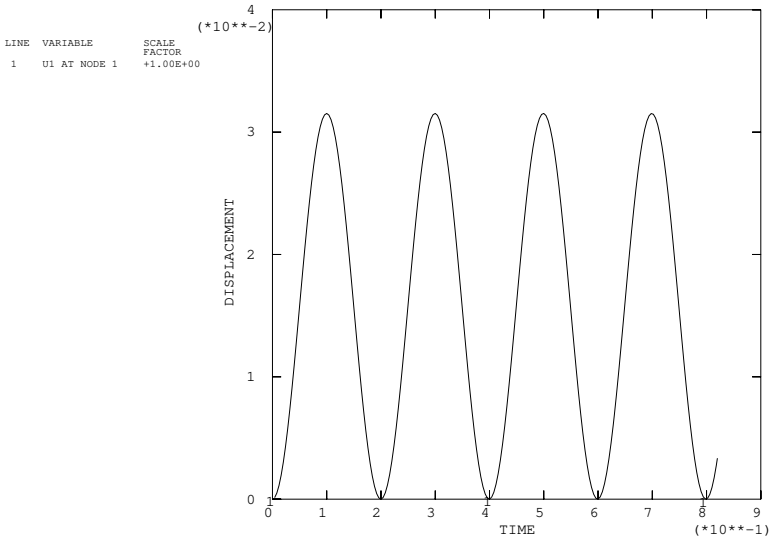


Figure 3: Displacement history; tabular, periodic, or equally spaced amplitude definition.

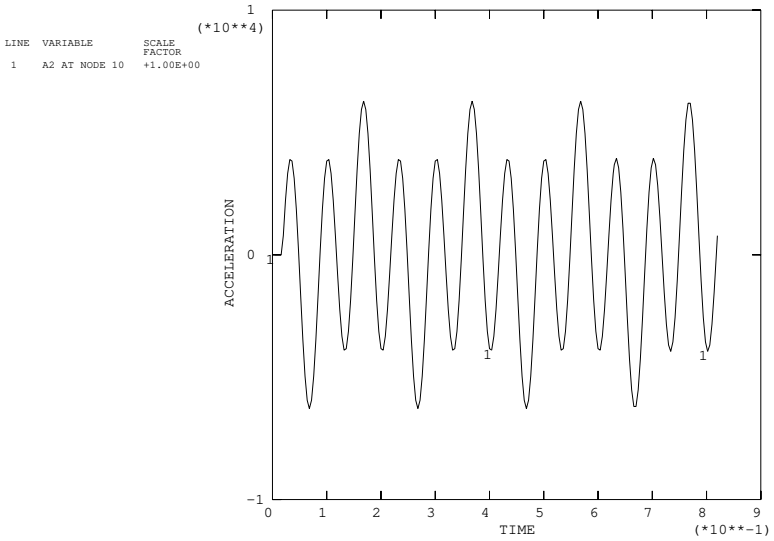


Figure 4: Acceleration history; modulated amplitude definition.

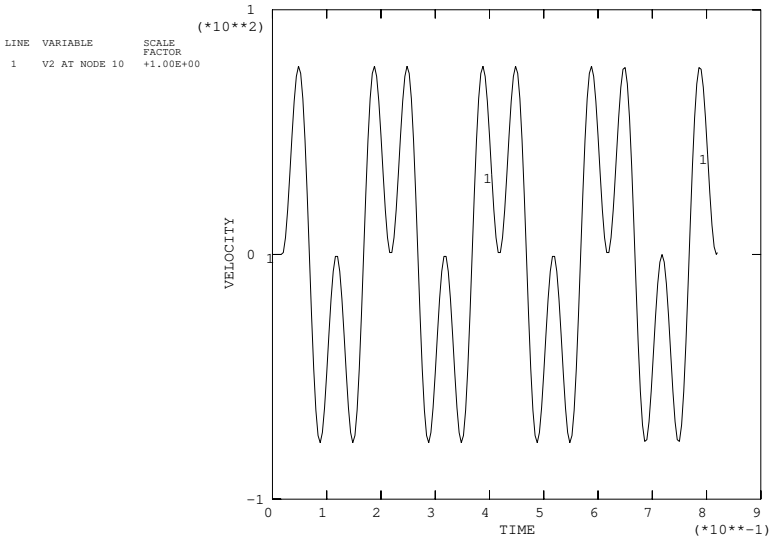


Figure 5: Velocity history; modulated amplitude definition.

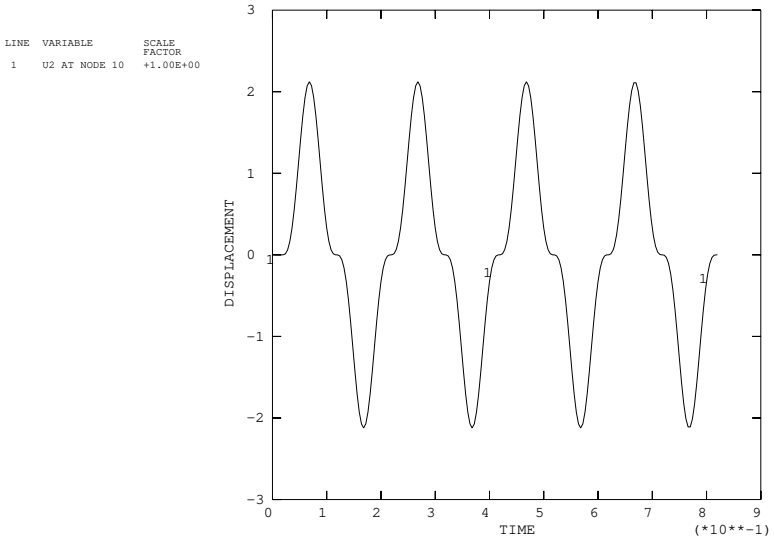


Figure 6: Displacement history; modulated amplitude definition.

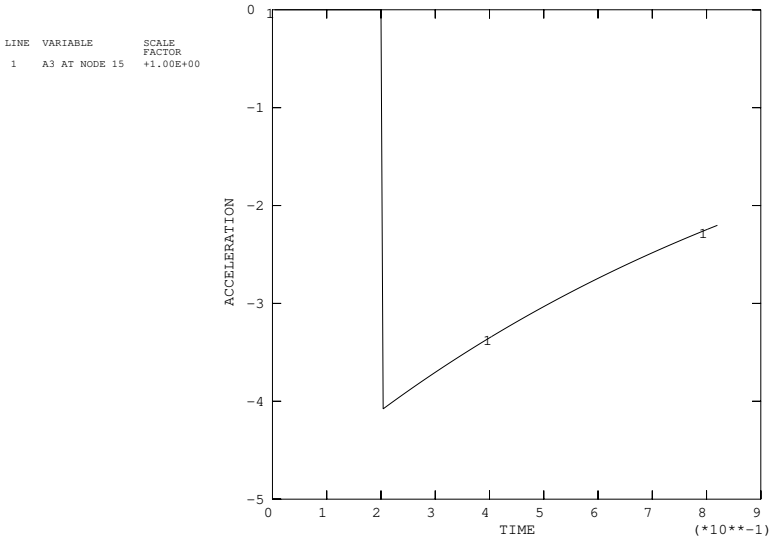


Figure 7: Acceleration history; exponential decay amplitude definition.

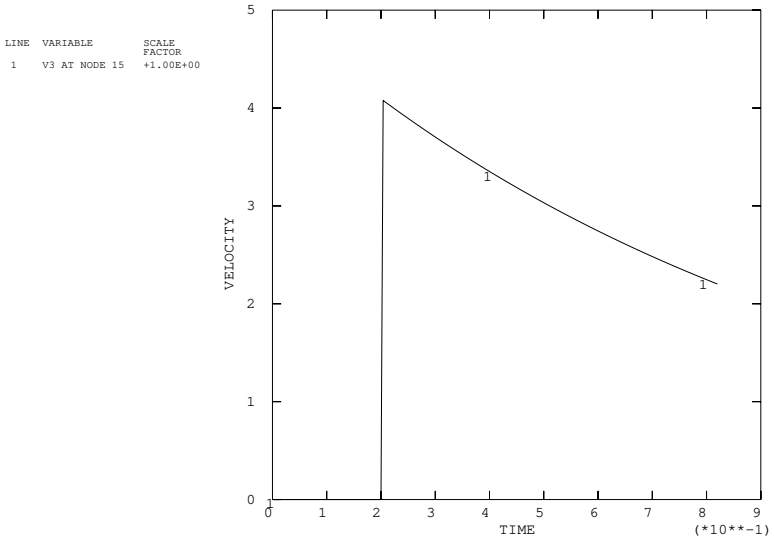


Figure 8: Velocity history; exponential decay amplitude definition.

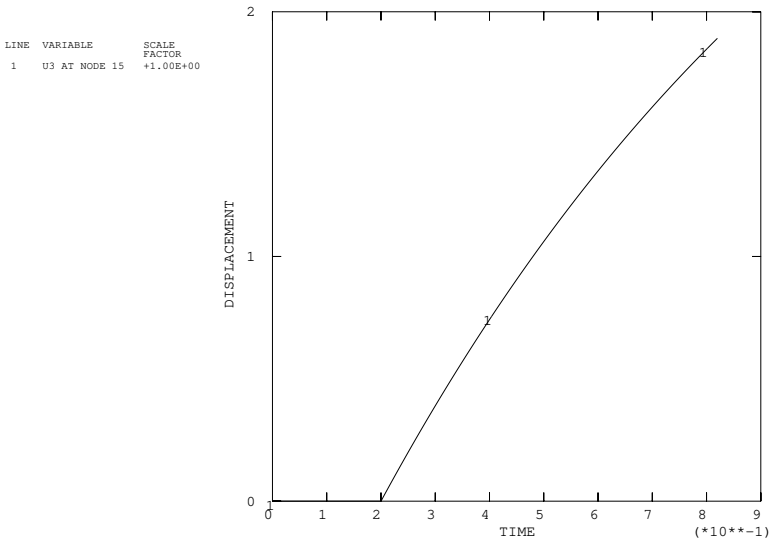


Figure 9: Displacement history; exponential decay amplitude definition.

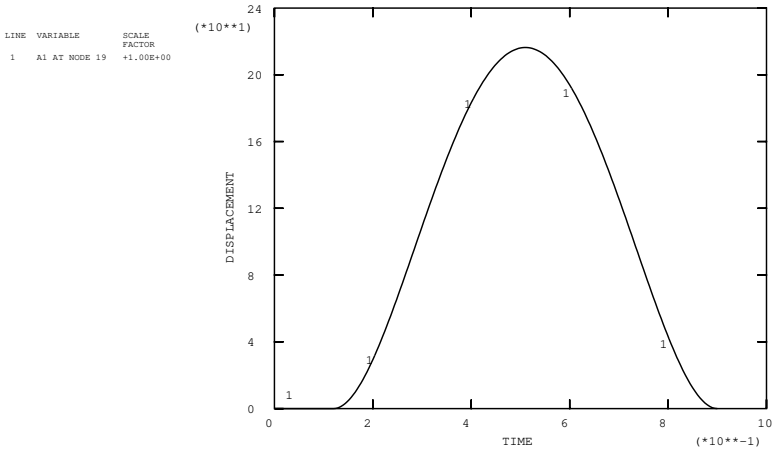


Figure 10: Acceleration history; smooth-step amplitude definition.

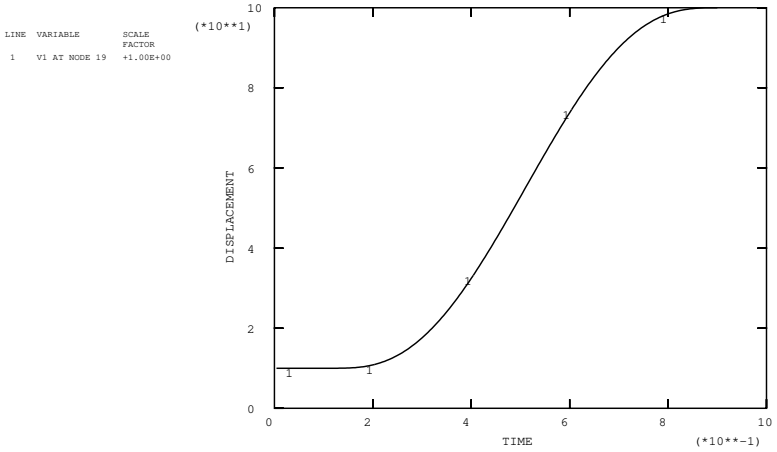


Figure 11: Velocity history; smooth-step amplitude definition.

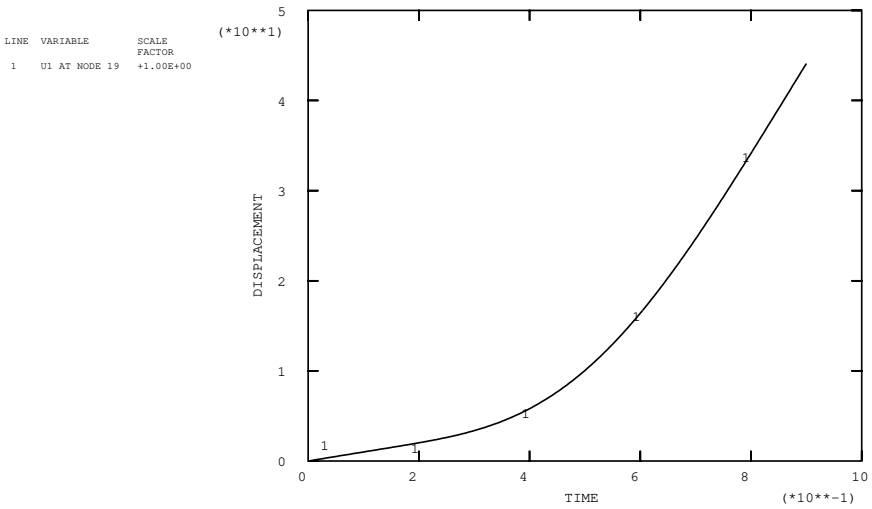


Figure 12: Displacement history; smooth-step amplitude definition.

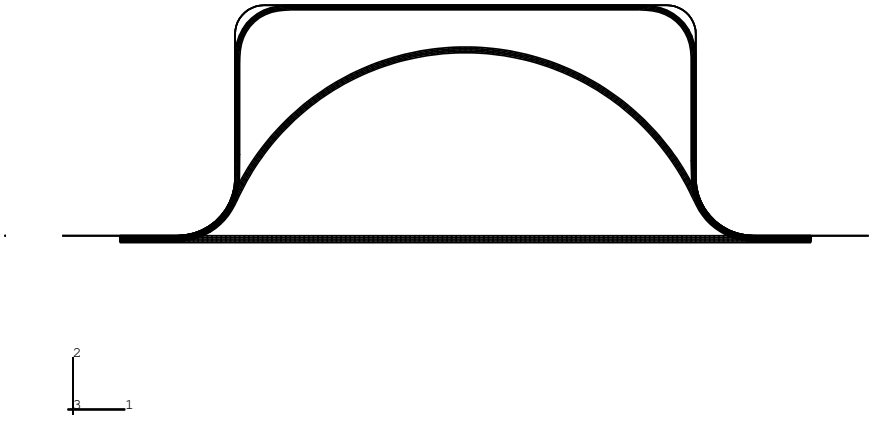


Figure 13: Configurations in original run; solution-dependent amplitude definition.

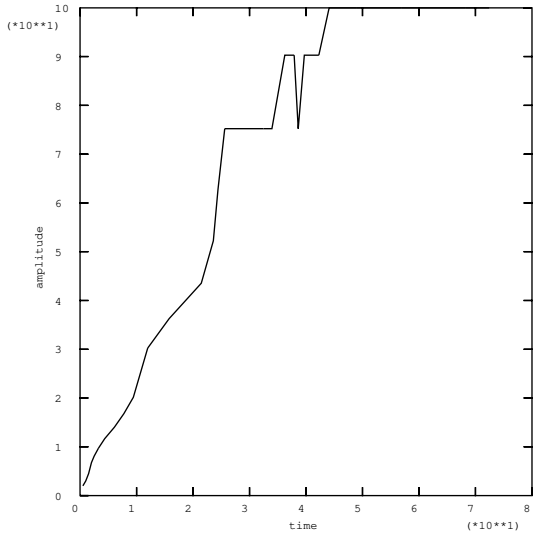


Figure 14: Amplitude history of original run; solution-dependent amplitude definition.

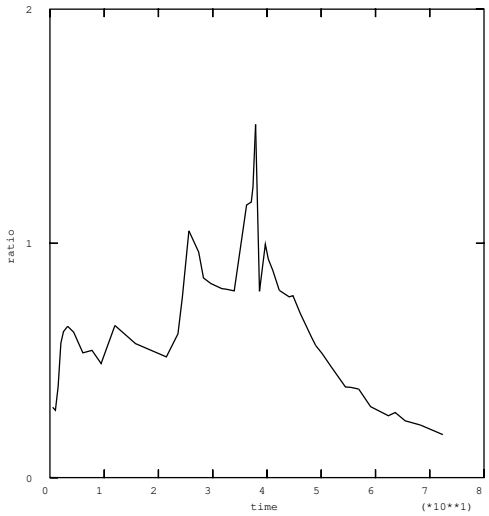


Figure 15: Ratio history of original run; solution-dependent amplitude definition.

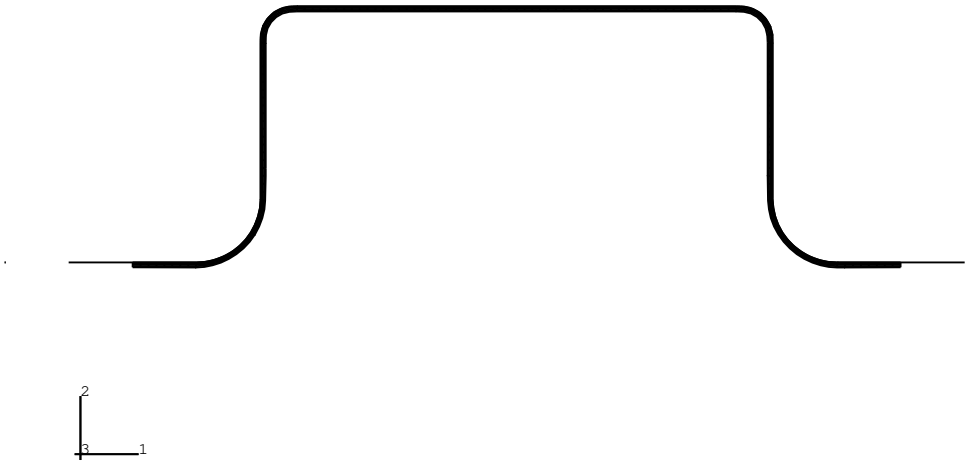


Figure 16: Final configuration after restart; solution-dependent amplitude definition.

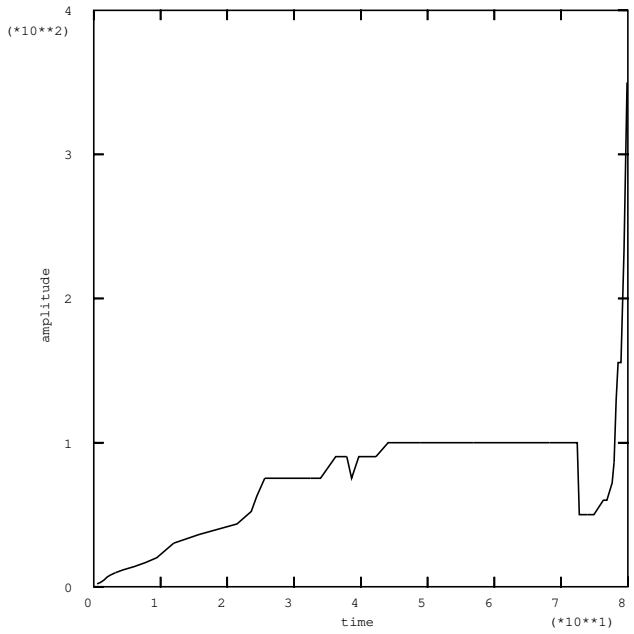


Figure 17: Total amplitude history; solution-dependent amplitude definition.

Spatially varying element properties

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Features tested

This section demonstrates the use of distributions to model spatially varying element properties.

Using distributions to define thicknesses, offsets, material orientations, and stiffness matrices for shell and membrane elements

Elements tested

S3R STRI3 S4 S4R S4R5 STRI65 S8R S8R5 S9R5
SC6R SC8R
SAX1 SAX2 SAXA
M3D3 M3D4 M3D4R M3D6 M3D8 M3D8R M3D9 M3D9R

Problem description

The analyses in this section demonstrate how distributions can be used to define spatially varying element properties in shells and membrane elements (the membrane tests apply only to Abaqus/Standard).

The geometry in each shell test is a flat plate modeled with either 9 quadrilateral shell elements or 18 triangular shell elements. In most test cases each shell element is assigned a different thickness, offset, and material orientation using distributions. In some cases both distributions and nodal thicknesses are used to define the shell thicknesses. A linear elastic orthotropic material is used in each case. All the test cases in this section are verified by creating equivalent reference models using multiple section assignments to define the shell thicknesses, offsets, and material properties. Some of these reference models are included.

The geometry in each membrane test is a flat plate modeled with either 9 quadrilateral membrane elements or 18 triangular membrane elements. In most test cases each membrane element is assigned a different thickness and material orientation using distributions. A linear elastic material is used in each case. Initial stresses are applied to the membrane elements in all tests. All the test cases in this section are verified by creating equivalent reference models using multiple section assignments to define the membrane thicknesses and material orientations.

Loading: The multistep Abaqus/Standard analysis performed on each shell model consists of the following:

- Step 1: A frequency analysis.
- Step 2: A steady-state dynamic analysis with modal damping and nodal loads.
- Step 3: A modal dynamic analysis with modal damping and nodal loads.
- Step 4: A direct steady-state dynamic analysis with modal damping and nodal loads.
- Step 5: A subspace projection steady-state dynamic analysis with nodal loads.
- Step 6: A random response analysis with nodal loads.

- Step 7: A response spectrum analysis.
- Step 8: A geometrically nonlinear static analysis with displacement boundary conditions.
- Step 9: A load case static analysis using distributed body loads, gravity loads, and centrifugal loads.

The membrane tests are identical to the shell tests except that an initial static step is included to resolve the initial stresses.

A single geometrically nonlinear dynamic step with displacement boundary conditions is used for the shell models testing Abaqus/Explicit.

Results and discussion

The results for each model agree with the associated reference solutions.

Input files

Abaqus/Standard analyses

distrib_multistep_s3r_gs_std.inp	Multistep analysis using S3R elements and a homogeneous general shell section definition. Distributions are used to define shell thicknesses, offsets, and material orientations.
distrib_multistep_s3r_nt_gs_std.inp	Multistep analysis using S3R elements, a homogeneous general shell section definition, and nodal thicknesses. Distributions are used to define shell thicknesses and material orientations.
distrib_multistep_s3r_std.inp	Multistep analysis using S3R elements with a homogeneous shell section definition that is integrated during the analysis. Distributions are used to define shell thicknesses, offsets, and material orientations.
distrib_multistep_s3r_nt_std.inp	Multistep analysis using S3R elements with a homogeneous shell section definition that is integrated during the analysis and nodal thicknesses. Distributions are used to define shell thicknesses and material orientations.
distrib_multistep_s4_gs_std.inp	Multistep analysis using S4 elements and a homogeneous general shell section. Distributions are used to define shell thicknesses, offsets, and material orientations.
distrib_multistep_s4_gs_ref_std.inp	Reference solution for distrib_multistep_s4_gs_std.inp using multiple homogeneous shell section definitions to define varying shell thicknesses, offsets, and material orientations.
distrib_multistep_s4_nt_gs_std.inp	Multistep analysis using S4 elements, a homogeneous general shell section definition, and nodal thicknesses. Distributions are used to define shell thicknesses and material orientations.
distrib_multistep_s4_std.inp	Multistep analysis using S4 elements with a homogeneous shell section definition that is integrated during the analysis. Distributions are used to define shell thicknesses, offsets, and material orientations.
distrib_multistep_s4_rs_std.inp	Restart analysis for distrib_multistep_s4_std.inp.
distrib_multistep_s4_nt_std.inp	Multistep analysis using S4 elements with a homogeneous shell section definition that is integrated during the analysis and nodal thicknesses. Distributions are used to define shell thicknesses and material orientations.
distrib_multistep_s4_nt_ref_std.inp	Reference solution for distrib_multistep_s4_nt_std.inp using multiple homogeneous shell section definitions and nodal thicknesses to define varying shell thicknesses, offsets, and material orientations.

distrib_multistep_s4_nt1_gs_ std.inp	Multistep analysis using S4 elements, a homogeneous general shell section definition, and nodal thicknesses. Distributions are used to define offsets and material orientations.
distrib_multistep_s4_nt1_std.inp	Multistep analysis using S4 elements with a homogeneous shell section definition that is integrated during the analysis and nodal thicknesses. Distributions are used to define offsets and material orientations.
distrib_multistep_s4_st_std.inp	Multistep analysis using S4 elements with a homogeneous general shell section definition. The section stiffness is specified directly. Distributions are used to define shell section stiffness and material orientations. The distributions for the section stiffnesses and material orientations are read from separate input files distrib_s4_stiff_in.inp and distrib_s4_orient_in.inp.
distrib_s4_stiff_in.inp	Input file for section stiffness distribution for distrib_multistep_s4_st_std.inp.
distrib_s4_orient_in.inp	Input file for material orientation distribution for distrib_multistep_s4_st_std.inp.
distrib_multistep_s4r_gs_std.inp	Multistep analysis using S4R elements and a homogeneous general shell section definition. Distributions are used to define shell thicknesses, offsets, and material orientations.
distrib_multistep_s4r_std.inp	Multistep analysis using S4R elements with a homogeneous shell section definition that is integrated during the analysis. Distributions are used to define shell thicknesses, offsets, and material orientations.
distrib_multistep_s4r_hyp_std.inp	Multistep analysis using S4R elements with a homogeneous shell section definition that is integrated during the analysis and a hyperelastic material. Distributions are used to define shell thicknesses, offsets, and material orientations.
distrib_multistep_s4r5_gs_std.inp	Multistep analysis using S4R5 elements and a homogeneous general shell section definition. Distributions are used to define shell thicknesses, offsets, and material orientations.
distrib_multistep_s4r5_std.inp	Multistep analysis using S4R5 elements with a homogeneous shell section definition that is integrated during the analysis. Distributions are used to define shell thicknesses, offsets, and material orientations.
distrib_multistep_s8r_gs_std.inp	Multistep analysis using S8R elements and a homogeneous general shell section definition. Distributions are used to define shell thicknesses, offsets, and material orientations.
distrib_multistep_s8r_std.inp	Multistep analysis using S8R elements with a homogeneous shell section definition that is integrated during the analysis. Distributions are used to define shell thicknesses, offsets, and material orientations.
distrib_multistep_s8r5_gs_std.inp	Multistep analysis using S8R5 elements and a homogeneous general shell section definition. Distributions are used to define shell thicknesses, offsets, and material orientations.
distrib_multistep_s8r5_std.inp	Multistep analysis using S8R5 elements with a homogeneous shell section definition that is integrated during the analysis. Distributions are used to define shell thicknesses, offsets, and material orientations.
distrib_multistep_s9r5_gs_std.inp	Multistep analysis using S9R5 elements and a homogeneous general shell section definition. Distributions are used to define shell thicknesses, offsets, and material orientations.
distrib_multistep_s9r5_std.inp	Multistep analysis using S9R5 elements with a homogeneous shell section definition that is integrated during the analysis. Distributions are used to define shell thicknesses, offsets, and material orientations.
distrib_multistep_sc6r_gs_std.inp	Multistep analysis using SC6R elements and a homogeneous general shell section definition. Distributions are used to define material orientations.

distrib_multistep_sc6r_std.inp	Multistep analysis using SC6R elements with a homogeneous shell section definition that is integrated during the analysis. Distributions are used to define material orientations.
distrib_multistep_sc8r_gs_std.inp	Multistep analysis using SC8R elements and a homogeneous general shell section definition. Distributions are used to define material orientations.
distrib_multistep_sc8r_std.inp	Multistep analysis using SC8R elements with a homogeneous shell section definition that is integrated during the analysis. Distributions are used to define material orientations.
distrib_multistep_stri3_gs_std.inp	Multistep analysis using STRI3 elements and a homogeneous general shell section definition. Distributions are used to define shell thicknesses, offsets, and material orientations.
distrib_multistep_stri3_std.inp	Multistep analysis using STRI3 elements with a homogeneous shell section definition that is integrated during the analysis. Distributions are used to define shell thicknesses, offsets, and material orientations.
distrib_multistep_stri65_gs_ std.inp	Multistep analysis using STRI65 elements and a homogeneous general shell section definition. Distributions are used to define shell thicknesses, offsets, and material orientations.
distrib_multistep_stri65_std.inp	Multistep analysis using STRI65 elements with a homogeneous shell section definition that is integrated during the analysis. Distributions are used to define shell thicknesses, offsets, and material orientations.
distrib_multistep_sax1_std.inp	Multistep analysis using SAX1 elements with a homogeneous shell section definition that is integrated during the analysis. Distributions are used to define shell thicknesses and offsets.
distrib_multistep_sax2_std.inp	Multistep analysis using SAX2 elements with a homogeneous shell section definition that is integrated during the analysis. Distributions are used to define shell thicknesses and offsets.
distrib_multistep_sax2t_std.inp	Multistep analysis using SAX2T elements with a homogeneous shell section definition that is integrated during the analysis. Distributions are used to define shell thicknesses and offsets.
distrib_multistep_saxa12_std.inp	Multistep analysis using SAXA12 elements with a homogeneous shell section definition that is integrated during the analysis. Distributions are used to define shell thicknesses, offsets, and material orientations.
distrib_multistep_saxa22_std.inp	Multistep analysis using SAXA22 elements with a homogeneous shell section definition that is integrated during the analysis. Distributions are used to define shell thicknesses, offsets, and material orientations.
distrib_multistep_comp_s3r_gs_ std.inp	Multistep analysis using S3R elements and a composite general shell section definition. Distributions are used to define shell thicknesses, offsets, and material orientations.
distrib_multistep_comp_s3r_nt_ gs_std.inp	Multistep analysis using S3R elements, a composite general shell section definition, and nodal thicknesses. Distributions are used to define shell thicknesses and material orientations.
distrib_multistep_comp_s3r_ std.inp	Multistep analysis using S3R elements with a composite shell section definition that is integrated during the analysis. Distributions are used to define shell thicknesses, offsets, and material orientations.
distrib_multistep_comp_s3r_nt_ std.inp	Multistep analysis using S3R elements with a composite shell section definition that is integrated during the analysis and nodal thicknesses. Distributions are used to define shell thicknesses and material orientations.
distrib_multistep_comp_s3r_1_ std.inp	Multistep analysis using S3R elements with a composite shell section definition that is integrated during the analysis. Distributions are used to

	define shell thicknesses, offsets, material orientations, composite layer thicknesses, and composite layer angles.
distrib_multistep_comp_s4_gs_std.inp	Multistep analysis using S4 elements and a composite general shell section. Distributions are used to define shell thicknesses, offsets, and material orientations.
distrib_multistep_comp_s4_gs_ref_std.inp	Reference solution for distrib_multistep_comp_s4_gs_std.inp using multiple composite shell section definitions to define varying shell thicknesses, offsets, and material orientations.
distrib_multistep_comp_s4_nt_gs_std.inp	Multistep analysis using S4 elements, a composite general shell section definition, and nodal thicknesses. Distributions are used to define shell thicknesses and material orientations.
distrib_multistep_comp_s4_std.inp	Multistep analysis using S4 elements with a composite shell section definition that is integrated during the analysis. Distributions are used to define shell thicknesses, offsets, and material orientations.
distrib_multistep_comp_s4_rs_std.inp	Restart analysis for distrib_multistep_comp_s4_std.inp.
distrib_multistep_comp_s4_nt_std.inp	Multistep analysis using S4 elements with a composite shell section definition that is integrated during the analysis and nodal thicknesses. Distributions are used to define shell thicknesses and material orientations.
distrib_multistep_comp_s4_nt_ref_std.inp	Reference solution for distrib_multistep_comp_s4_nt_std.inp using multiple composite shell section definitions and nodal thicknesses to define varying shell thicknesses, offsets, and material orientations.
distrib_multistep_comp_s4_nt1_gs_std.inp	Multistep analysis using S4 elements, a composite general shell section definition, and nodal thicknesses. Distributions are used to define offsets and material orientations.
distrib_multistep_comp_s4_nt1_std.inp	Multistep analysis using S4 elements with a composite shell section definition that is integrated during the analysis and nodal thicknesses. Distributions are used to define offsets and material orientations.
distrib_multistep_comp_s4_3_std.inp	Multistep analysis using S4 elements with a composite shell section definition that is integrated during the analysis. Distributions are used to define shell thicknesses, offsets, material orientations, composite layer thicknesses, and composite layer angles.
distrib_multistep_comp_s4_4_std.inp	Multistep analysis using S4 elements with a composite shell section definition that is integrated during the analysis. Distributions are used to define shell thicknesses, offsets, material orientations, composite layer thicknesses, and composite layer angles.
distrib_multistep_comp_s4_5_std.inp	Multistep analysis using S4 elements with a composite shell section definition that is integrated during the analysis. Distributions are used to define shell thicknesses, offsets, material orientations, composite layer thicknesses, and composite layer angles.
distrib_multistep_comp_s4r_gs_std.inp	Multistep analysis using S4R elements and a composite general shell section definition. Distributions are used to define shell thicknesses, offsets, and material orientations.
distrib_multistep_comp_s4_gs_3_std.inp	Multistep analysis using S4 elements with a composite general shell section definition. Distributions are used to define shell thicknesses, offsets, material orientations, composite layer thicknesses, and composite layer angles.
distrib_multistep_comp_s4_gs_4_std.inp	Multistep analysis using S4 elements with a composite general shell section definition. Distributions are used to define shell thicknesses, offsets, material orientations, composite layer thicknesses, and composite layer angles.

distrib_multistep_comp_s4_gs_5_ std.inp	Multistep analysis using S4 elements with a composite general shell section definition. Distributions are used to define shell thicknesses, offsets, material orientations, composite layer thicknesses, and composite layer angles.
distrib_multistep_comp_s4r_ std.inp	Multistep analysis using S4R elements with a composite shell section definition that is integrated during the analysis. Distributions are used to define shell thicknesses, offsets, and material orientations.
distrib_multistep_comp_s4r_hyp_ std.inp	Multistep analysis using S4R elements with a composite shell section definition that is integrated during the analysis and a hyperelastic material. Distributions are used to define shell thicknesses, offsets, and material orientations.
distrib_multistep_comp_s4r5_gs_ std.inp	Multistep analysis using S4R5 elements and a composite general shell section definition. Distributions are used to define shell thicknesses, offsets, and material orientations.
distrib_multistep_comp_s4r5_ std.inp	Multistep analysis using S4R5 elements with a composite shell section definition that is integrated during the analysis. Distributions are used to define shell thicknesses, offsets, and material orientations.
distrib_multistep_comp_s8r_gs_ std.inp	Multistep analysis using S8R elements and a composite general shell section definition. Distributions are used to define shell thicknesses, offsets, and material orientations.
distrib_multistep_comp_s8r_ std.inp	Multistep analysis using S8R elements with a composite shell section definition that is integrated during the analysis. Distributions are used to define shell thicknesses, offsets, and material orientations.
distrib_multistep_comp_s8r5_gs_ std.inp	Multistep analysis using S8R5 elements and a composite general shell section definition. Distributions are used to define shell thicknesses, offsets, and material orientations.
distrib_multistep_comp_s8r5_ std.inp	Multistep analysis using S8R5 elements with a composite shell section definition that is integrated during the analysis. Distributions are used to define shell thicknesses, offsets, material orientations, composite layer thicknesses, and composite layer angles.
distrib_multistep_comp_s9r5_gs_ std.inp	Multistep analysis using S9R5 elements and a composite general shell section definition. Distributions are used to define shell thicknesses, offsets, and material orientations.
distrib_multistep_comp_s9r5_ std.inp	Multistep analysis using S9R5 elements with a composite shell section definition that is integrated during the analysis. Distributions are used to define shell thicknesses, offsets, material orientations, composite layer thicknesses, and composite layer angles.
distrib_multistep_comp_sc6r_gs_ std.inp	Multistep analysis using SC6R elements and a composite general shell section definition. Distributions are used to define material orientations.
distrib_multistep_comp_sc6r_ std.inp	Multistep analysis using SC6R elements with a composite shell section definition that is integrated during the analysis. Distributions are used to define material orientations.
distrib_multistep_comp_sc8r_gs_ std.inp	Multistep analysis using SC8R elements and a composite general shell section definition. Distributions are used to define material orientations.
distrib_multistep_comp_sc8r_ std.inp	Multistep analysis using SC8R elements with a composite shell section definition that is integrated during the analysis. Distributions are used to define material orientations.
distrib_multistep_comp_sc8r_1_ std.inp	Multistep analysis using SC8R elements with a composite shell section definition that is integrated during the analysis. Distributions are used to define material orientations and composite layer orientation angles.

distrib_multistep_comp_sc8r_2_std.inp	Multistep analysis using SC8R elements with a composite shell section definition that is integrated during the analysis. Distributions are used to define material orientations and composite layer orientation angles.
distrib_multistep_comp_sc8r_gs_1_std.inp	Multistep analysis using SC8R elements with a composite general shell section definition. Distributions are used to define material orientations and composite layer orientation angles.
distrib_multistep_comp_sc8r_gs_2_std.inp	Multistep analysis using SC8R elements with a composite general shell section definition. Distributions are used to define material orientations and composite layer orientation angles.
distrib_multistep_comp_stri3_gs_std.inp	Multistep analysis using STRI3 elements and a composite general shell section definition. Distributions are used to define shell thicknesses, offsets, and material orientations.
distrib_multistep_comp_stri3_std.inp	Multistep analysis using STRI3 elements with a composite shell section definition that is integrated during the analysis. Distributions are used to define shell thicknesses, offsets, material orientations, composite layer thicknesses, and composite layer angles.
distrib_multistep_comp_stri65_gs_std.inp	Multistep analysis using STRI65 elements and a composite general shell section definition. Distributions are used to define shell thicknesses, offsets, and material orientations.
distrib_multistep_comp_stri65_std.inp	Multistep analysis using STRI65 elements with a composite shell section definition that is integrated during the analysis. Distributions are used to define shell thicknesses, offsets, and material orientations.
distrib_multistep_comp_saxa12_std.inp	Multistep analysis using SAXA12 elements with a composite shell section definition that is integrated during the analysis. Distributions are used to define shell thicknesses, offsets, and material orientations.
distrib_multistep_comp_saxa22_std.inp	Multistep analysis using SAXA22 elements with a composite shell section definition that is integrated during the analysis. Distributions are used to define shell thicknesses, offsets, material orientations, composite layer thicknesses, and composite layer angles.
distrib_multistep_m3d3_std.inp	Multistep analysis using M3D3 elements. Distributions are used to define membrane thicknesses and material orientations.
distrib_multistep_m3d4_std.inp	Multistep analysis using M3D4 elements. Distributions are used to define membrane thicknesses and material orientations.
distrib_multistep_m3d4r_std.inp	Multistep analysis using M3D4R elements. Distributions are used to define membrane thicknesses and material orientations.
distrib_multistep_m3d6_std.inp	Multistep analysis using M3D6 elements. Distributions are used to define membrane thicknesses and material orientations.
distrib_multistep_m3d8_std.inp	Multistep analysis using M3D8 elements. Distributions are used to define membrane thicknesses and material orientations.
distrib_multistep_m3d8r_std.inp	Multistep analysis using M3D8R elements. Distributions are used to define membrane thicknesses and material orientations.
distrib_multistep_m3d9_std.inp	Multistep analysis using M3D9 elements. Distributions are used to define membrane thicknesses and material orientations.
distrib_multistep_m3d9r_std.inp	Multistep analysis using M3D9R elements. Distributions are used to define membrane thicknesses and material orientations.

Abaqus/Explicit analyses

distrib_s4_nt_xpl.inp	Analysis using S4 elements with a shell section definition that is integrated during the analysis and nodal thicknesses. Distributions are used to define shell thicknesses and material orientations.
distrib_s4_nt_ref_xpl.inp	Reference solution for distrib_s4_nt_xpl.inp using multiple shell section definitions and nodal thicknesses to define varying shell thicknesses and material orientations.
distrib_s4r_nt_xpl.inp	Analysis using S4R elements with a shell section definition that is integrated during the analysis and nodal thicknesses. Distributions are used to define shell thicknesses and material orientations.
distrib_s4r_nt_ref_xpl.inp	Reference solution for distrib_s4r_nt_xpl.inp using multiple shell section definitions and nodal thicknesses to define varying shell thicknesses and material orientations.
distrib_comp_s4_nt_xpl.inp	Multistep analysis using S4 elements with a composite shell section definition that is integrated during the analysis. Distributions are used to define shell thicknesses and material orientations.
distrib_comp_s4_nt_ref_xpl.inp	Reference solution for distrib_comp_s4_nt_xpl.inp using multiple composite shell section definitions and nodal thicknesses to define varying shell thicknesses and material orientations.
distrib_comp_s3r_angle_xpl.inp	Analysis using S3R elements with composite shell section definitions that are integrated during the analysis. Distributions are used to define composite layer angles and thicknesses.
distrib_comp_s3r_angle_ref_ xpl.inp	Reference solution for distrib_comp_s3r_angle_xpl.inp using multiple composite shell section definitions to define varying composite layer angles and thicknesses.
distrib_comp_s4_angle_xpl.inp	Analysis using S4 elements with composite shell section definitions that are integrated during the analysis. Distributions are used to define composite layer angles and thicknesses.
distrib_comp_s4_angle_ref_ xpl.inp	Reference solution for distrib_comp_s4_angle_xpl.inp using multiple composite shell section definitions to define varying composite layer angles and thicknesses.
distrib_comp_s4r_angle_xpl.inp	Analysis using S4R elements with composite shell section definitions that are integrated during the analysis. Distributions are used to define composite layer angles and thicknesses.
distrib_comp_s4r_angle_ref_ xpl.inp	Reference solution for distrib_comp_s4r_angle_xpl.inp using multiple composite shell section definitions to define varying composite layer angles and thicknesses.
distrib_comp_s4rs_angle_xpl.inp	Analysis using S4RS elements with composite shell section definitions that are integrated during the analysis. Distributions are used to define composite layer angles and thicknesses.
distrib_comp_s4rs_angle_ref_ xpl.inp	Reference solution for distrib_comp_s4rs_angle_xpl.inp using multiple composite shell section definitions to define varying composite layer angles and thicknesses.
distrib_comp_s3r_gs_angle_ xpl.inp	Analysis using S3R elements with composite general shell section definitions. Distributions are used to define composite layer angles and thicknesses.
distrib_comp_s3r_gs_angle_ref_ xpl.inp	Reference solution for distrib_comp_s3r_gs_angle_xpl.inp using multiple composite shell general section definitions to define varying composite layer angles and thicknesses.

distrib_comp_s4_gs_angle_xpl.inp	Analysis using S4 elements with composite general shell section definitions. Distributions are used to define composite layer angles and thicknesses.
distrib_comp_s4_gs_angle_ref_xpl.inp	Reference solution for distrib_comp_s4_gs_angle_xpl.inp using multiple composite shell general section definitions to define varying composite layer angles and thicknesses.
distrib_comp_s4r_gs_angle_xpl.inp	Analysis using S4R elements with composite general shell section definitions. Distributions are used to define composite layer angles and thicknesses.
distrib_comp_s4r_gs_angle_ref_xpl.inp	Reference solution for distrib_comp_s4r_gs_angle_xpl.inp using multiple composite shell general section definitions to define varying composite layer angles and thicknesses.
distrib_comp_s4rs_gs_angle_xpl.inp	Analysis using S4RS elements with composite general shell section definitions. Distributions are used to define composite layer angles and thicknesses.
distrib_comp_s4rs_gs_angle_ref_xpl.inp	Reference solution for distrib_comp_s4rs_gs_angle_xpl.inp using multiple composite shell general section definitions to define varying composite layer angles and thicknesses.
distrib_s3r_gs_stiff_xpl.inp	Analysis using S3R elements with homogeneous general shell section definitions and direct section stiffness specification. Distributions are used to define shell section stiffness.
distrib_s3r_gs_stiff_ref_xpl.inp	Reference solution for distrib_s3r_gs_stiff_xpl.inp using multiple homogeneous general shell section definitions to define varying shell section stiffness.
distrib_s4_gs_stiff_xpl.inp	Analysis using S4 elements with homogeneous general shell section definitions and direct section stiffness specification. Distributions are used to define shell section stiffness.
distrib_s4_gs_stiff_ref_xpl.inp	Reference solution for distrib_s4_gs_stiff_xpl.inp using multiple homogeneous general shell section definitions to define varying shell section stiffness.
distrib_s4r_gs_stiff_xpl.inp	Analysis using S4R elements with homogeneous general shell section definitions and direct section stiffness specification. Distributions are used to define shell section stiffness.
distrib_s4r_gs_stiff_ref_xpl.inp	Reference solution for distrib_s4r_gs_stiff_xpl.inp using multiple homogeneous general shell section definitions to define varying shell section stiffness.
distrib_s4rs_gs_stiff_xpl.inp	Analysis using S4RS elements with homogeneous general shell section definitions and direct section stiffness specification. Distributions are used to define shell section stiffness.
distrib_s4rs_gs_stiff_ref_xpl.inp	Reference solution for distrib_s4rs_gs_stiff_xpl.inp using multiple homogeneous general shell section definitions to define varying shell section stiffness.

Using distributions to define thicknesses, material orientations, and material behaviors for continuum elements

Elements tested

CPS3 CPE3 CPS4 CPS4R CPE4 CPE4H CPE4I CPE4R
CPS6 CPS6M CPE6 CPE6M CPS8 CPS8R CPE8 CPE8R
CAX3 CAX4 CAX4H CAX4I CAX4R CAX6 CAX6M CAX8 CAX8R
CGAX3 CGAX4 CGAX4H CGAX4R CGAX6 CGAX6M CGAX8 CGAX8R
C3D4 C3D6 C3D8 C3D10 C3D10HS C3D10M C3D15 C3D20
CCL12 CCL24

MAX1 MAX2 MCL6 MCL9 MGAX1 MGAX2
CSS8

Problem description

The analyses in this section demonstrate how distributions can be used to define material orientations and material behavior on an element-by-element basis for continuum elements. The geometry in the two-dimensional tests is a unit square modeled with either 9 quadrilateral or 18 triangular elements. The geometry in the three-dimensional tests is a unit cube with between 8 to 12 elements. In most test cases each solid element is assigned a different material orientation using a distribution. In some of the test cases distributions of material behaviors are used. All the test cases in this section are verified by creating equivalent reference models using multiple section assignments to define material orientations. Some of these reference models are included. In some cases the residual mode functionality is also tested.

Some of the Abaqus/Standard tests include membrane elements with thicknesses and material orientations defined with distributions.

Loading: The multistep Abaqus/Standard analysis performed on each model consists of the following:

- Step 1: A frequency analysis.
- Step 2: A steady-state dynamic analysis with modal damping and nodal loads.
- Step 3: A modal dynamic analysis with modal damping and nodal loads.
- Step 4: A direct steady-state dynamic analysis with modal damping and nodal loads.
- Step 5: A subspace projection steady-state dynamic analysis with nodal loads.
- Step 6: A random response analysis with nodal loads.
- Step 7: A response spectrum analysis.
- Step 8: A geometrically nonlinear static analysis with displacement boundary conditions.
- Step 9: A load case static analysis using distributed body loads, gravity loads, and centrifugal loads.

For cases in which the residual mode functionality is tested, a static perturbation step is added prior to the frequency step.

Only Step 8 above is used to test the Abaqus/Explicit models.

Results and discussion

The results for each model agree with the associated reference solutions.

Input files

Abaqus/Standard analyses

distrib_multistep_c3d4_std.inp	Multistep analysis using C3D4 elements. Distributions are used to define material orientations. Residual modes are activated in the frequency step for use in the subsequent modal procedures.
distrib_multistep_c3d6_std.inp	Multistep analysis using C3D6 elements. Distributions are used to define material orientations. Residual modes are activated in the frequency step for use in the subsequent modal procedures.
distrib_multistep_comp_c3d6_std.inp	Multistep analysis using composite C3D6 elements. Distributions are used to define composite layer thicknesses and orientation angles.

distrib_multistep_c3d8_std.inp	Multistep analysis using C3D8 elements. Distributions are used to define material orientations and orthotropic elastic behavior. Residual modes are activated in the frequency step for use in the subsequent modal procedures.
distrib_multistep_c3d8_comp_std.inp	Multistep analysis using composite C3D8 elements. Distributions are used to define material orientations.
distrib_multistep_comp_c3d8_1_ std.inp	Multistep analysis using composite C3D8 elements. Distributions are used to define material orientations and composite layer orientation angles.
distrib_multistep_comp_c3d8_2_ std.inp	Multistep analysis using composite C3D8 elements. Distributions are used to define material orientations and composite layer orientation angles.
distrib_multistep_comp_c3d8_3_ std.inp	Multistep analysis using composite C3D8 elements. Distributions are used to define material orientations and composite layer orientation angles.
distrib_multistep_comp_c3d8_4_ std.inp	Multistep analysis using composite C3D8 elements. Distributions are used to define material orientations and composite layer orientation angles.
distrib_multistep_comp_c3d8_5_ std.inp	Multistep analysis using composite C3D8 elements. Distributions are used to define material orientations and composite layer orientation angles.
distrib_multistep_comp_c3d8_6_ std.inp	Multistep analysis using composite C3D8 elements. Distributions are used to define composite layer thicknesses and orientation angles.
distrib_multistep_c3d10_std.inp	Multistep analysis using C3D10 elements. Distributions are used to define material orientations, orthotropic elastic behavior, and material density.
distrib_multistep_c3d10hs_std.inp	Multistep analysis using C3D10HS elements. Distributions are used to define material orientations, orthotropic elastic behavior, and material density.
distrib_multistep_c3d10_ref_std.inp	Reference solution for distrib_multistep_c3d10_std.inp using multiple solid section and material definitions to define varying material orientations and material behavior.
distrib_multistep_c3d10hs_ref_std.inp	Reference solution for distrib_multistep_c3d10hs_std.inp using multiple solid section and material definitions to define varying material orientations and material behavior.
distrib_multistep_c3d10m_std.inp	Multistep analysis using C3D10M elements. Distributions are used to define material orientations and anisotropic elastic behavior. Residual modes are activated in the frequency step for use in the subsequent modal procedures.
distrib_multistep_c3d15_std.inp	Multistep analysis using C3D15 elements. Distributions are used to define material orientations.
distrib_multistep_comp_c3d15_ std.inp	Multistep analysis using composite C3D15 elements. Distributions are used to define composite layer thicknesses and orientation angles.
distrib_multistep_c3d20r_std.inp	Multistep analysis using C3D20R elements. Distributions are used to define material orientations. Residual modes are activated in the frequency step for use in the subsequent modal procedures.
distrib_multistep_c3d20r_comp1_ std.inp	Multistep analysis using composite C3D20R elements. Distributions are used to define material orientations.
distrib_multistep_c3d20r_comp2_ std.inp	Multistep analysis using composite C3D20R elements. Distributions are used to define material orientations.
distrib_multistep_ccl12_mcl6_std.inp	Multistep analysis using CCL12 and MCL6 elements. Distributions are used to define material orientations and membrane thicknesses.
distrib_multistep_ccl24_mcl9_std.inp	Multistep analysis using CCL24 and MCL9 elements. Distributions are used to define material orientations and membrane thicknesses.
distrib_multistep_cax3_std.inp	Multistep analysis using CAX3 elements. Distributions are used to define material orientations.

distrib_multistep_cax4_std.inp	Multistep analysis using CAX4 and MAX1 elements. Distributions are used to define material orientations and membrane thicknesses.
distrib_multistep_cax4h_std.inp	Multistep analysis using CAX4H elements. Distributions are used to define material orientations.
distrib_multistep_cax4i_std.inp	Multistep analysis using CAX4I elements. Distributions are used to define material orientations. Residual modes are activated in the frequency step for use in the subsequent modal procedures.
distrib_multistep_cax4r_std.inp	Multistep analysis using CAX4R elements. Distributions are used to define material orientations.
distrib_multistep_cax6_std.inp	Multistep analysis using CAX6 elements. Distributions are used to define material orientations.
distrib_multistep_cax6m_std.inp	Multistep analysis using CAX6M elements. Distributions are used to define material orientations.
distrib_multistep_cax8_std.inp	Multistep analysis using CAX8 and MAX2 elements. Distributions are used to define material orientations and membrane thicknesses.
distrib_multistep_cax8r_std.inp	Multistep analysis using CAX8R elements. Distributions are used to define material orientations. Residual modes are activated in the frequency step for use in the subsequent modal procedures.
distrib_multistep_cgax3_std.inp	Multistep analysis using CGAX3 elements. Distributions are used to define material orientations.
distrib_multistep_cgax4_std.inp	Multistep analysis using CGAX4 and MGAX1 elements. Distributions are used to define material orientations and membrane thicknesses.
distrib_multistep_cgax4h_std.inp	Multistep analysis using CGAX4H elements. Distributions are used to define material orientations.
distrib_multistep_cgax4r_std.inp	Multistep analysis using CGAX4R elements. Distributions are used to define material orientations.
distrib_multistep_cgax6_std.inp	Multistep analysis using CGAX6 elements. Distributions are used to define material orientations.
distrib_multistep_cgax8_std.inp	Multistep analysis using CGAX8 and MGAX2 elements. Distributions are used to define material orientations and membrane thicknesses.
distrib_multistep_cgax8r_std.inp	Multistep analysis using CGAX8R elements. Distributions are used to define material orientations.
distrib_multistep_cpe3_std.inp	Multistep analysis using CPE3 elements. Distributions are used to define material orientations.
distrib_multistep_cpe4_std.inp	Multistep analysis using CPE4 elements. Distributions are used to define material orientations.
distrib_multistep_cpe4h_std.inp	Multistep analysis using CPE4H elements. Distributions are used to define material orientations.
distrib_multistep_cpe4i_std.inp	Multistep analysis using CPE4I elements. Distributions are used to define material orientations.
distrib_multistep_cpe4r_std.inp	Multistep analysis using CPE4R elements. Distributions are used to define material orientations. Residual modes are activated in the frequency step for use in the subsequent modal procedures.
distrib_multistep_cpe6_std.inp	Multistep analysis using CPE6 elements. Distributions are used to define material orientations.
distrib_multistep_cpe6m_std.inp	Multistep analysis using CPE6M elements. Distributions are used to define material orientations.

distrib_multistep_cpe8_std.inp	Multistep analysis using CPE8 elements. Distributions are used to define material orientations, orthotropic elastic behavior (with engineering constants), and material density. Residual modes are activated in the frequency step for use in the subsequent modal procedures.
distrib_multistep_cpe8_ref_std.inp	Reference solution for distrib_multistep_cpe8_std.inp using multiple solid section definitions to define varying material orientations.
distrib_multistep_cpe8r_std.inp	Multistep analysis using CPE8R elements. Distributions are used to define material orientations.
distrib_multistep_cps3_std.inp	Multistep analysis using CPS3 elements. Distributions are used to define material orientations.
distrib_multistep_cps4_std.inp	Multistep analysis using CPS4 elements. Distributions are used to define material orientations and lamina elastic behavior.
distrib_multistep_cps4r_std.inp	Multistep analysis using CPS4R elements. Distributions are used to define material orientations.
distrib_multistep_cps6_std.inp	Multistep analysis using CPS6 elements. Distributions are used to define material orientations. Residual modes are activated in the frequency step for use in the subsequent modal procedures.
distrib_multistep_cps6m_std.inp	Multistep analysis using CPS6M elements. Distributions are used to define material orientations.
distrib_multistep_cps8_std.inp	Multistep analysis using CPS8 elements. Distributions are used to define material orientations.
distrib_multistep_cps8r_std.inp	Multistep analysis using CPS8R elements. Distributions are used to define material orientations. Residual modes are activated in the frequency step for use in the subsequent modal procedures.
distrib_multistep_css8_comp_std.inp	Multistep analysis using composite CSS8 elements. Distributions are used to define material orientations.
distrib_multistep_css8_comp_ref_std.inp	Reference solution for `distrib_multistep_css8_comp_std.inp` using multiple solid section definitions to define varying material orientations.
distrib_multistep_css8_std.inp	Multistep analysis using CSS8 elements. Distributions are used to define material orientations.
distrib_multistep_css8_ref_std.inp	Reference solution for `distrib_multistep_css8_std.inp` using multiple solid section definitions to define varying material orientations.

Abaqus/Explicit analyses

distrib_c3d10m_xpl.inp	Analysis using C3D10M elements. Distributions are used to define material orientations.
distrib_c3d10m_ref_xpl.inp	Reference solution for distrib_c3d10m_xpl.inp using multiple solid section definitions to define varying material orientations.
distrib_c3d10_xpl.inp	Analysis using C3D10 elements. Distributions are used to define material orientations.
distrib_c3d10_ref_xpl.inp	Reference solution for distrib_c3d10_xpl.inp using multiple solid section definitions to define varying material orientations.

Using distributions while transferring results from one Abaqus/Standard analysis to another

Elements tested

S3R S4R SAX1
CPE4 CPS4 C3D8

Problem description

The analyses in this section demonstrate that element properties defined with distributions can be transferred from one Abaqus/Standard analysis to another. All the test cases in this section are verified by creating equivalent reference models using multiple section assignments to define the shell thicknesses, offsets, and material properties. Some of these reference models are included.

Results and discussion

The results for each model agree with the associated reference solutions.

Input files

ss1_c3d8_ep.inp	Two geometrically nonlinear static steps using C3D8 elements. Distributions are used to define material orientations and orthotropic elastic behavior (using engineering constants).
ss2_c3d8_ep_n_n.inp	Imports both elements in ss1_c3d8_ep.inp at the end of Step 1 with UPDATE=NO and STATE=NO. One new C3D8 element is defined.
ss2_c3d8_ep_n_y.inp	Imports both elements in ss1_c3d8_ep.inp at the end of Step 1 with UPDATE=NO and STATE=YES. One new C3D8 element is defined.
ss2_c3d8_ep_y_n.inp	Imports both elements in ss1_c3d8_ep.inp at the end of Step 1 with UPDATE=YES and STATE=NO. One new C3D8 element is defined.
ss2_c3d8_ep_y_y.inp	Imports both elements in ss1_c3d8_ep.inp at the end of Step 1 with UPDATE=YES and STATE=YES. One new C3D8 element is defined.
ss1_c3d8_1_ep.inp	Two geometrically nonlinear static steps using C3D8 elements. Distributions are used to define material orientations and orthotropic elastic behavior.
ss2_c3d8_1_ep_n_n.inp	Imports both elements in ss1_c3d8_1_ep.inp at the end of Step 1 with UPDATE=NO and STATE=NO. One new C3D8 element is defined.
ss2_c3d8_1_ep_n_y.inp	Imports both elements in ss1_c3d8_1_ep.inp at the end of Step 1 with UPDATE=NO and STATE=YES. One new C3D8 element is defined.
ss2_c3d8_1_ep_y_n.inp	Imports both elements in ss1_c3d8_1_ep.inp at the end of Step 1 with UPDATE=YES and STATE=NO. One new C3D8 element is defined.
ss2_c3d8_1_ep_y_y.inp	Imports both elements in ss1_c3d8_1_ep.inp at the end of Step 1 with UPDATE=YES and STATE=YES. One new C3D8 element is defined.
ss1_c3d8_2_ep.inp	Two geometrically nonlinear static steps using C3D8 elements. Distributions are used to define material orientations and anisotropic elastic behavior.
ss2_c3d8_2_ep_n_n.inp	Imports both elements in ss1_c3d8_2_ep.inp at the end of Step 1 with UPDATE=NO and STATE=NO. One new C3D8 element is defined.
ss2_c3d8_2_ep_n_y.inp	Imports both elements in ss1_c3d8_2_ep.inp at the end of Step 1 with UPDATE=NO and STATE=YES. One new C3D8 element is defined.

ss2_c3d8_2_ep_y_n.inp	Imports both elements in ss1_c3d8_2_ep.inp at the end of Step 1 with UPDATE=YES and STATE=NO. One new C3D8 element is defined.
ss2_c3d8_2_ep_y_y.inp	Imports both elements in ss1_c3d8_2_ep.inp at the end of Step 1 with UPDATE=YES and STATE=YES. One new C3D8 element is defined.
ss1_cpe4_ep.inp	Two geometrically nonlinear static steps using CPE4 elements. Distributions are used to define material orientations and isotropic elastic behavior.
ss2_cpe4_ep_n_n.inp	Imports both elements in ss1_cpe4_ep.inp at the end of Step 1 with UPDATE=NO and STATE=NO. One new CPE4 element is defined.
ss2_cpe4_ep_n_y.inp	Imports both elements in ss1_cpe4_ep.inp at the end of Step 1 with UPDATE=NO and STATE=YES. One new CPE4 element is defined.
ss2_cpe4_ep_y_n.inp	Imports both elements in ss1_cpe4_ep.inp at the end of Step 1 with UPDATE=YES and STATE=NO. One new CPE4 element is defined.
ss2_cpe4_ep_y_y.inp	Imports both elements in ss1_cps4_ep.inp at the end of Step 1 with UPDATE=YES and STATE=YES. One new CPE4 element is defined.
ss1_cps4_ep.inp	Two geometrically nonlinear static steps using CPS4 elements. Distributions are used to define material orientations and lamina elastic behavior.
ss2_cps4_ep_n_n.inp	Imports both elements in ss1_cps4_ep.inp at the end of Step 1 with UPDATE=NO and STATE=NO. One new CPS4 element is defined.
ss2_cps4_ep_n_y.inp	Imports both elements in ss1_cps4_ep.inp at the end of Step 1 with UPDATE=NO and STATE=YES. One new CPS4 element is defined.
ss2_cps4_ep_y_n.inp	Imports both elements in ss1_cps4_ep.inp at the end of Step 1 with UPDATE=YES and STATE=NO. One new CPS4 element is defined.
ss2_cps4_ep_y_y.inp	Imports both elements in ss1_cps4_ep.inp at the end of Step 1 with UPDATE=YES and STATE=YES. One new CPS4 element is defined.
ss1_s3r_ep.inp	Two geometrically nonlinear static steps using S3R elements with a shell section definition that is integrated during the analysis. Distributions are used to define shell thicknesses, offsets, and material orientations.
ss2_s3r_ep_n_n.inp	Imports both elements in ss1_s3r_ep.inp at the end of Step 1 with UPDATE=NO and STATE=NO. One new S3R element is defined.
ss2_s3r_ep_n_y.inp	Imports both elements in ss1_s3r_ep.inp at the end of Step 1 with UPDATE=NO and STATE=YES. One new S3R element is defined.
ss2_s3r_ep_y_n.inp	Imports both elements in ss1_s3r_ep.inp at the end of Step 1 with UPDATE=YES and STATE=NO. One new S3R element is defined.
ss2_s3r_ep_y_y.inp	Imports both elements in ss1_s3r_ep.inp at the end of Step 1 with UPDATE=YES and STATE=YES. One new S3R element is defined.
ss1_s4r_ep.inp	Two geometrically nonlinear static steps using S4R elements with a shell section definition that is integrated during the analysis. Distributions are used to define shell thicknesses, offsets, and material orientations.
ss2_s4r_ep_n_n.inp	Imports both elements in ss1_s4r_ep.inp at the end of Step 1 with UPDATE=NO and STATE=NO. One new S4R element is defined.
ss2_s4r_ep_n_y.inp	Imports both elements in ss1_s4r_ep.inp at the end of Step 1 with UPDATE=NO and STATE=YES. One new S4R element is defined.
ss2_s4r_ep_y_n.inp	Imports both elements in ss1_s4r_ep.inp at the end of Step 1 with UPDATE=YES and STATE=NO. One new S4R element is defined.
ss2_s4r_ep_y_y.inp	Imports both elements in ss1_s4r_ep.inp at the end of Step 1 with UPDATE=YES and STATE=YES. One new S4R element is defined.
ss1_s4r_ep_ref.inp	Reference solution for ss1_s4r_ep.inp using multiple shell section definitions to define varying shell thicknesses, offsets, and material orientations.

ss2_s4r_ep_ref_n_n.inp	Reference solution for ss2_s4r_ep_n_n.inp.
ss2_s4r_ep_ref_n_y.inp	Reference solution for ss2_s4r_ep_n_y.inp.
ss2_s4r_ep_ref_y_n.inp	Reference solution for ss2_s4r_ep_y_n.inp.
ss2_s4r_ep_ref_y_y.inp	Reference solution for ss2_s4r_ep_y_y.inp.
ss1_s4r_ep_gs.inp	Two geometrically nonlinear static steps using S4R elements with a general shell section definition. Distributions are used to define shell thicknesses, offsets, and material orientations.
ss2_s4r_ep_gs_n_n.inp	Imports both elements in ss1_s4r_ep_gs.inp at the end of Step 1 with UPDATE=NO and STATE=NO. One new S4R element is defined.
ss2_s4r_ep_gs_n_y.inp	Imports both elements in ss1_s4r_ep_gs.inp at the end of Step 1 with UPDATE=NO and STATE=YES. One new S4R element is defined.
ss2_s4r_ep_gs_y_n.inp	Imports both elements in ss1_s4r_ep_gs.inp at the end of Step 1 with UPDATE=YES and STATE=NO. One new S4R element is defined.
ss2_s4r_ep_gs_y_y.inp	Imports both elements in ss1_s4r_ep_gs.inp at the end of Step 1 with UPDATE=YES and STATE=YES. One new S4R element is defined.
ss1_s4r_ep_gs_st.inp	Two geometrically nonlinear static steps using S4R elements with a general shell section definition. The section stiffness is specified directly. Distributions are used to define shell section stiffness and material orientations.
ss2_s4r_ep_gs_st_n_n.inp	Imports both elements in ss1_s4r_ep_gs_st.inp at the end of Step 1 with UPDATE=NO and STATE=NO. One new S4R element is defined.
ss2_s4r_ep_gs_st_n_y.inp	Imports both elements in ss1_s4r_ep_gs_st.inp at the end of Step 1 with UPDATE=NO and STATE=YES. One new S4R element is defined.
ss2_s4r_ep_gs_st_y_n.inp	Imports both elements in ss1_s4r_ep_gs_st.inp at the end of Step 1 with UPDATE=YES and STATE=NO. One new S4R element is defined.
ss2_s4r_ep_gs_st_y_y.inp	Imports both elements in ss1_s4r_ep_gs_st.inp at the end of Step 1 with UPDATE=YES and STATE=YES. One new S4R element is defined.
ss1_sax1_ep.inp	Two geometrically nonlinear static steps using SAX1 elements with a shell section definition that is integrated during the analysis. Distributions are used to define shell thicknesses and offsets.
ss2_sax1_ep_n_n.inp	Imports both elements in ss1_sax1_ep.inp at the end of Step 1 with UPDATE=NO and STATE=NO. One new SAX1 element is defined.
ss2_sax1_ep_n_y.inp	Imports both elements in ss1_sax1_ep.inp at the end of Step 1 with UPDATE=NO and STATE=YES. One new SAX1 element is defined.
ss2_sax1_ep_y_n.inp	Imports both elements in ss1_sax1_ep.inp at the end of Step 1 with UPDATE=YES and STATE=NO. One new SAX1 element is defined.
ss2_sax1_ep_y_y.inp	Imports both elements in ss1_sax1_ep.inp at the end of Step 1 with UPDATE=YES and STATE=YES. One new SAX1 element is defined.
ss1_sax1_ep_ref.inp	Reference solution for ss1_sax1_ep.inp using multiple shell section definitions to define varying shell thicknesses and offsets.
ss2_sax1_ep_ref_n_n.inp	Reference solution for ss2_sax1_ep_n_n.inp.
ss2_sax1_ep_ref_n_y.inp	Reference solution for ss2_sax1_ep_n_y.inp.
ss2_sax1_ep_ref_y_n.inp	Reference solution for ss2_sax1_ep_y_n.inp.
ss2_sax1_ep_ref_y_y.inp	Reference solution for ss2_sax1_ep_y_y.inp.

Using distributions for specifying orientation for connector elements

Elements tested

CONN3D2

Problem description

The analyses in this section demonstrate the use of distributions for specifying orientation for connectors. Some of these reference models are included.

Results and discussion

The results for each model agree with the associated reference solutions.

Input files

distrib_ori_connect_1.inp	Several CONN3D2 elements with beam-type connector sections are used in this test. Distributions are used to define orientations at the two nodes for the connector elements. All degrees of freedom are fixed at node a of the connector elements, and the **CLOAD* option is applied at node b such that the connector total force CTF in component 1 is 1.0 units, the total force in component 2 is 2.0 units, and the total force in component 3 is 3.0 units.
distrib_ori_connect_1_xpl.inp	Several CONN3D2 elements with beam-type connector sections are used in this test. Distributions are used to define orientations at the two nodes for the connector elements. All degrees of freedom are fixed at node a of the connector elements, and the **CLOAD* option is applied quasi-statically at node b such that the connector total force CTF in component 1 is 1.0 units, the total force in component 2 is 2.0 units, and the total force in component 3 is 3.0 units.

Define a part instance that will be used for display only

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Features tested

Definition of an instance used by Abaqus for display only without affecting the results of the rest of the model.

Problem description

The tests contain two instances, one of which is defined to be used for display only. This test verifies that the instance is not included in the analysis. In addition, it verifies the cases where the definition of the instance references zero, one, or three nodes from the other instance.

Results and discussion

These tests verify that the defined instance does not take part in the analysis.

Input files

displaybody_ref0.inp	Test with no reference nodes.
displaybody_ref1.inp	Test with one reference node.
displaybody_ref3.inp	Test with three reference nodes.

Establishing geostatic equilibrium when the initial stress state is unknown

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

CAX4P CAX4PH CAX4RP CAX4RPH
CAX6MP CAX6MPH
CAX8P CAX8PH CAX8RP CAX8RPH
CAXA8P1 CAXA8RP1
COH2D4P COH3D6P COH3D8P COHAX4P
CPE4P CPE4PH CPE4RP CPE4RPH
CPE6MP CPE6MPH
CPE8P CPE8RP
C3D8P C3D8PH C3D8RP C3D8RPH
C3D10MP C3D10MPH
C3D20P C3D20PH C3D20RP C3D20RPH

Features tested

Establishing geostatic equilibrium when the initial stress state is unknown is tested with various elements and materials.

Problem description

These tests verify the ability to establish geostatic equilibrium when the inital stress state is unknown for various combinations of materials and elements. Simple one-element tests are used in which pore pressure and distributed loads are applied.

Results and discussion

In all cases the results indicate that this option performs as expected. The absolute values of maximum displacements in all cases are within the limits specified. In addition, the results are close or identical to the results obtained without invoking automatic time incrementation, which is expected in these cases.

Input files

geo_nodisp_cax4p.inp	Elements CAX4P, CAX4RP, CAX4PH, and CAX4RPH with elastic and porous elastic materials; uniaxial compression; linearly varying pore pressure values prescribed.
geo_nodisp_cax6p.inp	Elements CAX6MP and CAX6MPH with elastic and porous elastic materials; uniaxial compression; linearly varying pore pressure values prescribed.
geo_nodisp_cax8p.inp	Elements CAX8P, CAX8RP, CAX8PH, and CAX8RPH with elastic and porous elastic materials; uniaxial compression; linearly varying pore pressure values prescribed.

geo_nodisp_caxa8p1.inp	Elements CAXA8P1 and CAXA8RP1 with elastic and porous materials; uniaxial compression; linearly varying pore pressure values prescribed.
geo_nodisp_coh2d.inp	Element COH2D4P with elastic material; uniaxial compression; linearly varying pore pressure values prescribed.
geo_nodisp_coh3d6p.inp	Element COH3D6P with elastic material; uniaxial compression; linearly varying pore pressure values prescribed.
geo_nodisp_coh3d8p.inp	Element COH3D8P with elastic material; uniaxial compression; linearly varying pore pressure values prescribed.
geo_nodisp_cohax4p.inp	Element COHAX4P with elastic material; uniaxial compression; linearly varying pore pressure values prescribed.
geo_nodisp_cpe4p.inp	Elements CPE4P, CPE4RP, CPE4PH, and CPE4RPH with elastic and porous elastic materials; uniaxial compression; linearly varying pore pressure values prescribed.
geo_nodisp_cpe4p_pla.inp	Elements CPE4P, CPE4RP, CPE4PH, and CPE4RPH with elastic material, extended Cam-clay plasticity model and Mohr-Coulomb model; uniaxial compression; zero pore pressure assigned at all nodes.
geo_nodisp_cpe4p_res.inp	Restart test for elements CPE4P, CPE4RP, CPE4PH, and CPE4RPH with elastic and porous elastic materials; uniaxial compression; linearly varying pore pressure values prescribed.
geo_nodisp_cpe6p.inp	Elements CPE6MP and CPE6MPH with elastic and porous elastic materials; uniaxial compression; linearly varying pore pressure values prescribed.
geo_nodisp_cpe8p.inp	Elements CPE8P, CPE8RP, CPE8PH, and CPE8RPH with elastic and porous elastic materials; uniaxial compression; linearly varying pore pressure values prescribed.
geo_nodisp_c3d8p.inp	Elements C3D8P, C3D8RP, C3D8PH, and C3D8RPH with elastic and porous elastic materials; uniaxial compression; linearly varying pore pressure values prescribed.
geo_nodisp_c3d10mp.inp	Elements C3D10MP and C3D10MPH with elastic and porous elastic materials; uniaxial compression; linearly varying pore pressure values prescribed.
geo_nodisp_c3d20p.inp	Elements C3D20P, C3D20RP, C3D20PH, and C3D20RPH with elastic and porous elastic materials; uniaxial compression; linearly varying pore pressure values prescribed.

Specifying geometric imperfection and parameter shape variation

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

CAX4H S4R S4R5

Features tested

Various methods of defining a geometric imperfection are tested. The same problems are performed using the parameter shape variation in Abaqus/Standard.

Problem description

The verification problems contained in this section test the geometric imperfection and parameter shape variation procedures in Abaqus. Simple geometries are used to test the various methods of defining an imperfection: specifying imperfection values, defining the imperfection as a linear superposition of eigenmodes, or using the results of a static analysis.

Results and discussion

The nodal coordinates of the perturbed geometry are verified to be correct.

Input files

Abaqus/Standard analyses

ximpa.inp	Imperfection specified as perturbation values in the cylindrical coordinate system.
xpsva.inp	Shape variation specified as variation values in the cylindrical coordinate system.
ximpb1.inp	Eigenvalue analysis of cylindrical shell structure.
ximpb2.inp	Imperfection defined by linear superposition of eigenmodes obtained from ximpb1.inp.
xpsvb2.inp	Shape variation defined by linear superposition of eigenmodes obtained from ximpb1.inp.
ximpc1.inp	Static analysis of a contact problem.
ximpc2.inp	Imperfection defined from the static analysis of problem ximpc1.inp.
xpsvc2.inp	Shape variation defined from the static analysis of problem ximpc1.inp.
ximpd.inp	File containing node numbers and coordinate perturbations at those nodes.
ximpe.inp	Imperfection specified as perturbation values in cartesian coordinates read from the file ximpd.inp.
ximpf.inp	Imperfection specified on the data lines as perturbation values in a spherical coordinate system.

Abaqus/Explicit analyses

imp_file1.inp	Static analysis performed with Abaqus/Standard.
imp_file2.inp	Imperfection defined from the static analysis of problem imp_file1.inp.
imp_rect_data.inp	File containing node numbers and coordinate perturbations at those nodes.
imp_rect.inp	Imperfection specified as perturbation values in Cartesian coordinates read from the file imp_rect_data.inp.
imp_spher.inp	Imperfection specified on data lines as perturbation values in the spherical coordinate system.

Specifying user-defined field variables directly in the input file

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Features tested

Direct method to redefine a field variable at a material point by associating it with a scalar output variable.

Problem description

In the direct approach, the appropriate field values are set equal to values of a scalar output variable from the previous increment automatically at run time. The tests verify that field variable values are transferred properly to a structure when the values are redefined at run time in the above manner.

Results and discussion

The results obtained using the direct approach are compared with the corresponding results from an identical model that utilizes user subroutine *USDFLD* to redefine field variables during an increment.

Input files

Abaqus/Standard analyses

plastic_udfld_specified_s.inp	An elastic-plastic material that has a built-in dependency on a solution variable (the equivalent plastic strain) is implemented with field variable dependence. In addition, the elastic-plastic hardening behavior is enhanced to include different hardening in tension versus compression. The latter utilizes field variable dependence, with the field variable associated with the hydrostatic pressure.
plastic_udfld_user_s.inp	The same setup as `plastic_udfld_specified_s.inp`, except that the field variables are redefined by user subroutine *USDFLD*.
plastic_udfld_user_s.f	User subroutine *USDFLD* used in `plastic_udfld_user_s.inp`.
udfcd1ht_sp.inp	Transient heat transfer analysis, DC1D2 elements. This model uses the **USER DEFINED FIELD*, TYPE=SPECIFIED option.
udfpbxsf_sp.inp	Static analysis followed by frequency analysis, elastic-plastic, B21 elements. This model uses the **USER DEFINED FIELD*, TYPE=SPECIFIED option.
udfpsxsf_sp.inp	Static analysis followed by frequency analysis, elastic-plastic, S4/S4R elements. This model uses the **USER DEFINED FIELD*, TYPE=SPECIFIED option.

Abaqus/Explicit analyses

plastic_udfld_specified_x.inp	An elastic-plastic material that has a built-in dependency on a solution variable (the equivalent plastic strain) is implemented with field variable dependence. In addition, the elastic-plastic hardening behavior is enhanced to include different hardening in

tension versus compression. The latter utilizes field variable dependence, with the field variable associated with the hydrostatic pressure.

Inertia-based load balancing

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Features tested

The verification problems contained in this section cover the common use cases for inertia relief in Abaqus/Standard. Relatively simple configurations have been selected to demonstrate how inertia-based load balancing can be used in static and direct-integration implicit dynamic steps.

Automobile suspension component

The structure analyzed in this problem is an automobile suspension component modeled with beam elements. The model is loaded with concentrated forces and moments at all free nodes. Inertia relief is used to find out if the applied loads are in equilibrium.

Static step

Elements tested

B31

Problem description

Model: The model consists of B31 elements with a circular cross-section configured to model the automobile A-arm.

Material: Density = 7800 kg/m^3, Young's modulus = 200× 10^9N/ m^2, Poisson's ratio = 0.3.

Boundary conditions: The model is fully constrained at node 3.

Loading: The model is loaded with concentrated forces and moments at all free nodes

Results and discussion

The analysis provides rigid body accelerations and corresponding inertia relief loads that balance the out-of-balance applied loads. The problem demonstrates how inertia relief can be used in place of a more expensive dynamic analysis to obtain constant rigid body accelerations.

Input files

irl_axle_b31.inp	Inertia relief for automobile suspension component.

Assembly loading and free body motion of a rocket under thrust

The problem models assembly loading and liftoff of a rocket. The inertia relief step provides the free body acceleration and static stresses due to the rocket thrust.

Static step

Elements tested

CAX4

Problem description

Model: The model consists of CAX4 elements with assembly loading modeled as a pre-tension bolt load. The thermal loading during lift-off and rocket thrust are modeled through internal and external pressures.

Material:

- Rocket: Density = 7800 kg/m^3, Young's modulus = 200× 10^9 N/ m^2, Poisson's ratio = 0.3.
- Engine: Density = 7000 kg/m^3, Young's modulus = 700× 10^7 N/ m^2, yield stress =380× 10^6 N/ m^2 .

Boundary conditions: The model is fixed at node 5 and has roller support at nodes 6, 7, and 8.

Loading:

- Step 1: A pre-tension section bolt loading is applied to simulate assembly loads, and a gravity load is applied for weight. These loads are propagated to the second and third steps.
- Step 2: Pressure loading to simulate thrust and thermal loads.
- Step 3: Inertia relief load.

Results and discussion

The results demonstrate how the inertia relief can be used for different types of loads and materials in a geometrically nonlinear analysis.

Input files

irl_rocket_cax4.inp	Inertia relief of a rocket at lift-off.

Stability analysis of a submerged structure (submarine)

This problem demonstrates how inertia relief can be used to establish initial static equilibrium when the external loads are not fully known.

Direct-integration implicit dynamic step

Elements tested

CPE4

Problem description

Model: The model consists of a longitudinal section of a submarine under gravity load and hydrostatic pressure at 52.5 m below sea level.

Material: Density = 7800 kg/m^3, Young's modulus = 200× 10^9 N/ m^2 , Poisson's ratio = 0.3, yield stress at 0 plastic strain = 380× 10^6 N/ m^2, yield stress at 0.35 plastic strain = 580× 10^6 N/ m^2.

Boundary conditions: No boundary conditions are applied in this model.

Loading: A transient dynamic procedure is used with the gravity load and hydrostatic pressure applied instantaneously, and a pressure load simulating shock-wave loading is ramped over the step.

Results and discussion

The results show that inertia relief can provide the unknown inertia forces that keep a submarine in equilibrium under static preload.

Input files

irl_stability_cpe4.inp	Stability analysis of a submerged structure.

Inertia relief with multiple load cases

This problem demonstrates how inertia relief can be used with multiple load cases.

Static step

Elements tested

CPE4

Problem description

Model: This problem consists of an airplane modeled as a free body with no boundary conditions. Multiple load cases are used to model various loading scenarios.

Material: Density = 7800 kg/m^3, Young's modulus = 200× 10^9 N/ m^2, Poisson's ratio = 0.3.

Boundary conditions: No boundary conditions are applied in the model.

Loading:

- Step 1: Multiple load cases are used to model various combinations of pressure loading with inertia relief loading.

Results and discussion

The results show that inertia relief can be used with multiple load cases to analyze various loading scenarios efficiently.

Input files

irl_multiload_cpe4.inp	Inertia relief with multiple load cases.

Inertia relief of a model with substructures

This problem demonstrates how inertia relief can be used with substructures in a geometrically linear analysis.

Static step

Elements tested
T2D2

Problem description

Model: The problem consist of an overhead hoist crane modeled using substructures. Each member is 1 m in length and 5 mm in diameter.

Material: Density = 7800 kg/m^3, Young's modulus = 200×10^9 N/ m^2, Poisson's ratio = 0.3.

Boundary conditions: The hoist is a simple pin-joined frame work that is constrained at the left end and mounted on rollers at the right end. The members can rotate freely at the joints.

Loading:

- Step 1: A concentrated load is applied at node 102.

Results and discussion

This analysis shows that inertia relief can be used with substructures.

Input files

irl_substruct_t2d2.inp	Overhead hoist model using substructures.
irl_sub_gen1.inp	Substructure generation file referenced in irl_substructure_t2d2.inp.
irl_sub_gen2.inp	Substructure generation file referenced in irl_substructure_t2d2.inp.

Defining the cutting surface

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

B21 B22 B31 B32 PIPE21 PIPE31
C3D4 C3D6 C3D8 C3D8T C3D20R C3D20RT C3D10M C3D10MT
CAX3 CAX4 CAX8 CAX6M
CPE4R CPE6M CPS3 CPS4 CPS8
DC3D4 DC3D6 DC3D8 DC3D10 DC3D15 DC3D20
GK3D8
M3D4R
R3D3 R3D4
S4R S8R S8RT SC8R
T2D2 T3D2

Features tested

The cutting surface is defined to create a cross-section-like surface by cutting an element set with a plane. The resulting surface is formed over the element facets, edges, or ends so as to be a close approximation to the specified plane. This interior surface can then be used for purposes such as pre-tensioning and requesting output of the force and the torque transmitted across that surface.

Problem description

These tests verify the creation of cross-section-like surfaces over various element types (continuum, structural, heat transfer, and rigid elements) by defining the cutting surface.

Results and discussion

The resulting surfaces are visually verified by postprocessing in the Visualization module of Abaqus/CAE.

Input files

cutting_plane_std_2d.inp	Generate surface over two-dimensional elements in Abaqus/Standard.
cutting_plane_std_3d.inp	Generate surface over three-dimensional elements in Abaqus/Standard.
cutting_plane_std_coupled.inp	Generate surface in an Abaqus/Standard coupled temperature-displacement analysis.
cutting_plane_std_thermal.inp	Generate surface in an Abaqus/Standard heat transfer analysis.
cutting_plane_std_submodel.inp	Generate surface in an Abaqus/Standard submodel analysis (global model: xptssi2.inp).
cutting_plane_std_submodel_sb.inp	Generate surface in an Abaqus/Standard stress-based submodel analysis (global model: xptssi2.inp).

cutting_plane_xpl_2d.inp	Generate surface over two-dimensional elements in Abaqus/Explicit.
cutting_plane_xpl_3d.inp	Generate surface over three-dimensional elements in Abaqus/Explicit.
cutting_plane_sx_s_preten.inp	Generate surface for use with pre-tensioning in Abaqus/Standard.
cutting_plane_sx_x_preten.inp	Generate surface in an Abaqus/Explicit analysis that is imported from Abaqus/Standard analysis cutting_plane_sx_s_preten.inp.

Defining a matrix for part of a model

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Features tested

This section contains tests for direct input of sparse matrices in Abaqus/Standard. Tests contain simple geometries with the static procedure.

Linear perturbation analysis of a truss model

A linear perturbation analysis is performed for a two-dimensional truss structure modeled with matrices.

Elements tested

T2D2

Problem description

Model: Some of the truss elements are replaced by sparse matrices representing stiffness.

Material: Young's modulus = 2.0×10^{11}, Poisson's ratio = 0.3.

Boundary conditions: The truss model is simply supported with a hinge support on one end and a roller support on the other end. The nodes with boundary conditions are part of the matrices.

Loading: Concentrated loads are applied at nodes that are either part of the matrices or shared between a matrix and an element.

Results and discussion

Displacements and loads from the matrix-based model are compared to the element-based model.

Input files

truss_matrix.inp	Truss model with matrix.

Multiple load case analysis of a beam model with an equation constraint and a multi-point constraint

A multiple load case analysis is performed for a two-dimensional beam model consisting of beam elements and matrices connected by kinematic constraints. For verification purposes, each load case is also analyzed in a separate step.

Elements tested

B22

Problem description

Model: Two beams, each consisting of one beam element and one matrix, are used. The first beam has a TIEMPC between a beam element node and a matrix node. The second beam has an equation constraint between a beam element node and a matrix node.

Material: Young's modulus = 2.81×10^7, Poisson's ratio = 0.3.

Boundary conditions: The beams are fixed at one end and free at the other end. The boundary conditions remain the same for all steps and load cases.

Loading: A concentrated load and moment are applied at the free end at a node that is part of the matrix for each beam. Each load is applied in a separate step and also as separate load cases in the multiple load case step.

Results and discussion

Results from the matrix-based model are compared to an element-based model for each load case.

Input files

mpceqn_matrix.inp	Beam model with **EQUATION* and **MPC* at matrix nodes.

Large-sliding contact with node-based contact surface

Large-sliding contact is simulated by moving a single two-dimensional continuum element represented by a matrix over other elements.

Elements tested

CPE4

Problem description

Model: The model contains two CPE4 elements and a matrix representing a CPE4 element. Contact is modeled with a node-based secondary surface on the matrix nodes and an element-based main surface over the continuum elements.

Material: Young's modulus = 3.0×10^7, Poisson's ratio = 0.0, friction coefficient = 0.1.

Boundary conditions: The continuum elements underlying the main surface are fully supported. Matrix nodes are pressed against the continuum element in the first step to simulate normal contact. In the second step, matrix nodes are moved tangent to the main surface to simulate large sliding.

Results and discussion

The displacement solution indicates that the contact constraints are satisfied exactly.

Input files

contact_matrix.inp	Large-sliding contact model with matrix and two-dimensional continuum elements.
contact_stiff.inp	Matrix representing stiffness for a CPE4 element.

Three-dimensional model with predefined temperatures and distributed surface loads

This problem demonstrates how to apply surface loads and predefined temperatures in matrix-based models.

Elements tested

C3D6

Problem description

Model: A cube is modeled with a C3D6 element and a matrix representing another C3D6 element. The element shares nodes with the matrix. Surface elements are defined on the matrix nodes to apply surface loads.

Material: Young's modulus = 3.0×10^6, Poisson's ratio = 0.3.

Boundary conditions: Boundary conditions are applied to all nodes in different directions.

Loading: Surface loads are applied to various faces of the cube. Predefined temperatures are applied for thermal straining.

Results and discussion

Surface loads over the matrix nodes give the same results as the element-based model. Predefined temperatures at nodes shared between the matrix and the element produce correct thermal strains in the element. No effect is observed on the matrix behavior due to predefined temperatures at the matrix nodes.

Input files

tempdsl_matrix.inp	Three-dimensional model with surface loads and predefined temperatures.
tempdsl_stiff.inp	Matrix representing the stiffness for the C3D6 element.

Tip loading of a diving board

A static analysis is performed with concentrated loads at the free end of a diving board.

Elements tested

B31 S4R

Problem description

Model: The diving board is modeled using shell elements. The support for the diving board consisting of shell and beam elements is replaced by a sparse stiffness matrix.

Material: Young's modulus = 3.0×10^7, Poisson's ratio = 0.29.

Boundary conditions: Nodes 5, 6, 7, 8, 70, 71, 72, 73, 210, and 213 (part of the matrix) are constrained in all six degrees of freedom.

Loading: The free end of the diving board is loaded with concentrated loads at the corner nodes.

Results and discussion

The analysis provides displacements for the diving board and reaction forces at the boundary nodes on the matrix. The results match those obtained from an element-based model.

Input files

divingboard_matrix.inp	Diving board with support modeled through matrix.
divingboard_stiff.inp	Matrix representing stiffness for diving board support.
divingboard_ele.inp	Diving board with support modeled using elements.

Natural frequency extraction of a diving board

A natural frequency analysis is performed.

Elements tested

B31 S4R

Problem description

Model: The diving board is modeled using shell elements. The support for the diving board consisting of shell and beam elements is replaced by sparse stiffness and mass matrices.

Material: Young's modulus = 3.0×10^7, Poisson's ratio = 0.29.

Boundary conditions: Nodes 5, 6, 7, 8, 70, 71, 72, 73, 210, and 213 (part of the matrix) are constrained in all six degrees of freedom.

Results and discussion

The analysis provides displacements for the diving board and reaction forces at the boundary nodes on the matrix. The results match those obtained from an element-based model.

Input files

divingboard_matrix_freq.inp	Diving board with support modeled through matrix.
divingboard_stiff.inp	Matrix representing stiffness for diving board support.
divingboard_mass.inp	Matrix representing mass for diving board support.
divingboard_ele_freq.inp	Diving board with support modeled using elements.

User-defined coordinate system

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

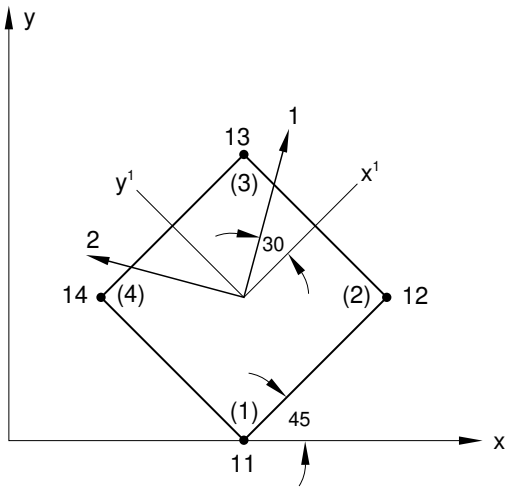
S4R

Features tested

The definition of local material axes by a user-defined coordinate system.

Problem description

In each of the tests a user-defined coordinate system is used to define the material point orientation as shown in *Figure 1*.



GLOBAL NODE	LOCAL NODE	COORDINATES
11	1	(1.0, 0.0, 0.0)
12	2	(1.7071, 0.7071, 0.0)
13	3	(1.0, 1.4142, 0.0)
14	4	(0.2929, 0.7071, 0.0)

Figure 1: Material point orientation.

Material: Linear elastic, Young's modulus = 3.0×10^6, Poisson's ratio = 0.3.

Boundary conditions: $u_x = u_y = u_z = 0$ at nodes 11 and 14, $\phi_z = 0$ for all the nodes.

Loading: Concentrated forces of 1000 are applied to nodes 12 and 13 at an angle of 45° to the x-axis.

Remarks

Local coordinate directions are used in the input file *xorisrdc.inp*.

Results and discussion

All tests should yield the following results:

Stress Components	Strain Components
σ_{11} = 1499.9	ε_{11} = 4.4997E−5
σ_{22} = 499.97	ε_{22} = 1.6665E−6
σ_{12} = −865.97	ε_{12} = −7.5050E−5

Principal Stresses	Principal Strains
σ_{max} = 2000.0	ε_{max} = 6.6662E−5
σ_{min} = 0.0	ε_{min} = −1.9998E−5

Input files

xorisrdc.inp

```
*ORIENTATION, NAME=LOCAL,
SYSTEM=RECTANGULAR,
DEFINITION=COORDINATES
2., 1.7071, 0., 1.,
1.4142, 0., 1., 0.7071, 0.
3, 30.
```

xoriscdc.inp

```
*ORIENTATION, NAME=LOCAL,
SYSTEM=CYLINDRICAL,
DEFINITION=COORDINATES
3.0, 0., 0., 3.0, 1.0, 0.0
2, -15.
```

xorissdc.inp

```
*ORIENTATION, NAME=LOCAL,
SYSTEM=SPHERICAL,
DEFINITION=COORDINATES
3., .7071068, 0., 3., 1., 0.
2, -15.
```

xoriszdc.inp

```
*ORIENTATION, NAME=LOCAL,
SYSTEM=ZRECTANGULAR,
DEFINITION=COORDINATES
0., 0., 1., 1., 1., 0.
3, 30.
```

xorisrdn.inp

```
*ORIENTATION, NAME=LOCAL,
SYSTEM=RECTANGULAR,
DEFINITION=NODES
12, 13, 11
3, 30.
```

xoriscdn.inp

```
*ORIENTATION, NAME=LOCAL,
SYSTEM=CYLINDRICAL,
DEFINITION=NODES
11, 12
2, 30.
```

xorissdn.inp

```
*ORIENTATION, NAME=LOCAL,
SYSTEM=SPHERICAL,
DEFINITION=NODES
12, 13
2, -15.
```

xoriszdn.inp

```
*ORIENTATION, NAME=LOCAL,
SYSTEM=ZRECTANGULAR,
DEFINITION=NODES
11, 12
2, 75.
```

xorisrdo.inp

```
*ORIENTATION, NAME=LOCAL,
SYSTEM=RECTANGULAR,
DEFINITION=OFFSET
2, 3
3, 30.
```

xoriscdo.inp

```
*ORIENTATION, NAME=LOCAL,
SYSTEM=CYLINDRICAL,
DEFINITION=OFFSET
1, 2
2, 30.
```

xorissdo.inp

```
*ORIENTATION, NAME=LOCAL,
SYSTEM=SPHERICAL,
DEFINITION=OFFSET
1, 2
3, 75.
```

xoriszdo.inp

```
*ORIENTATION, NAME=LOCAL,
SYSTEM=ZRECTANGULAR,
DEFINITION=OFFSET
2, 3
3, 30.
```

Pre-tension section

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

B21 B31
C3D6 C3D8 C3D8IH C3D8R C3D10 C3D10M C3D20 C3D27H
CAX4 CAX4RH CAX8R CAX4T
CPE3 CPE3H CPE4 CPE4R CPE8 CPE8RT
CPS4R CPS6M CPS8
T2D2 T3D2 T3D3

Features tested

Applying a prescribed assembly load on a variety of structures by means of a pre-tension section is tested.

Problem description

This set of tests verifies that the proper prescribed assembly load is applied to a structure using a pre-tension section. Loading is done by enforcing either a concentrated force (pre-tension load) or a displacement (tightening) at the pre-tension node (see *Prescribed Assembly Loads* for a description of this option). The structure is preloaded in the first step. In most cases it is further loaded in the second step, ensuring that the tightening is maintained.

The majority of the models are two-element meshes with boundary conditions that allow for uniform stretching of the cross-section. Thus, results verification is straightforward. Some input files have several two-element meshes with different element types set up in parallel.

In the first input file the total force in the defined pre-tension sections is output (see *Output to the Data and Results Files*). The total force results in the direction perpendicular to the sections match the reaction forces at the reference nodes associated with the pre-tension sections exactly.

The analyses include a submodel run (with the pre-tension section fully enclosed by the submodel boundary) and a substructure run (where the substructure's retained degrees of freedom belong to the pre-tension section).

Results and discussion

Analysis results indicate that the prescribed force or displacement is always established across the pre-tension section. Uniform sections yield a uniform axial stress given the analysis boundary conditions. Results after subsequent loading in the second step also indicate that the prescribed tightening of the section is maintained properly.

A full example that makes use of this feature is included in *Axisymmetric analysis of bolted pipe flange connections*.

Input files

xptssib2a.inp	Tests in parallel, multiple element types; **SECTION FILE* tests.

xptspit3.inp	Linear static perturbation.
xptsse23.inp	Tests in parallel, multiple element types.
xptssh2a3.inp	Hyperelastic material.
xptsri2.inp	Static analysis, Riks method.
xptsdib.inp	Dynamic analysis.
xptsdea.inp	Dynamic analysis with orthotropic material.
xptsdh2.inp	Direct-integration and subspace-based steady-state dynamics with hyperelastic material.
xptsti2a.inp	Coupled temperature-displacement.
xptsfit.inp	Natural frequency extraction with steady-state dynamics.
xptssi3.inp	C3D10 mixed with C3D20 elements; automatic midface node generation.
xptssh3.inp	**TRANSFORM* used on section nodes.
xptssi23.inp	User-defined section normal; degenerate elements.
xptssi2z.inp	Static analysis with substructures.
xptssi2.inp	Submodeling, global analysis.
xptssi2s.inp	Submodeling, local analysis.

Radiation view factor definition: symmetries and blocking

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Features tested

Radiation symmetry is verified in this test suite by comparing results obtained from models using the different symmetry options to the results obtained from the full model without symmetries. A few different configurations are used to allow the testing of all the symmetry options in two-dimensional, three-dimensional, and axisymmetric cases. Some of the configurations are also used to test radiation blocking.

Since the primary interest of this verification suite is the calculation of view factors in nontrivial geometries, all the problems consist of only a single increment in a single step of steady-state heat transfer analysis. No analytical solutions exist for the nontrivial configurations selected; therefore, verification of the results is limited to a comparison of variations of this problem, run with different types and levels of symmetry. All the results documented can be reproduced by running the input files provided with the Abaqus release.

Infinitely long square section tube

Two-dimensional models

Elements tested

DC2D4

Problem description

Four different two-dimensional models of the cross-section of the square tube are used: the full model, a half model with one reflection symmetry, a quarter model with two reflection symmetries, and a quarter model with cyclic symmetry. The full, half, and quarter models are shown in *Figure 1*. The two-dimensional models imply that the tube extends infinitely in the direction normal to the cross-section.

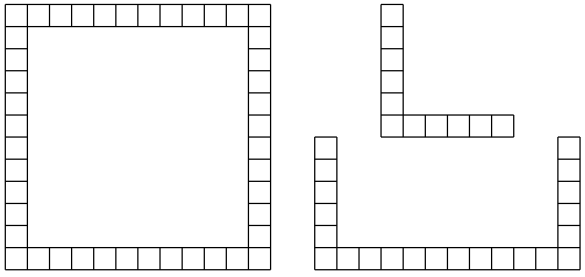


Figure 1: Two-dimensional square tube models.

Results and discussion

	Element 6, Side 3		
	RADFL	VFTOT	FTEMP
xrv24sn000.inp	3823.0	1.0	517.7
xrv24snr10.inp	3823.0	1.0	517.7
xrv24snr20.inp	3823.0	1.0	517.7
xrv24snc04.inp	3823.0	1.0	517.7

	Element 21, Side 2		
	RADFL	VFTOT	FTEMP
xrv24sn000.inp	–4787.0	1.0	719.5
xrv24snr10.inp	–4787.0	1.0	719.5
xrv24snr20.inp	–4787.0	1.0	719.5
xrv24snc04.inp	–4787.0	1.0	719.5

Input files

xrv24sn000.inp	Full model, DC2D4 elements.
xrv24snr10.inp	Half model, DC2D4 elements, one reflection symmetry.
xrv24snr20.inp	Quarter model, DC2D4 elements, two reflection symmetries.
xrv24snc04.inp	Quarter model, DC2D4 elements, cyclic symmetry (NC=4).

Three-dimensional models

Elements tested

DC3D8

Problem description

Three different models of the square section tube are used. In all cases the complete cross-section is modeled, and the infinite extent of the tube is simulated by using periodic symmetry in the direction normal to the cross-section of the tube. The three models differ in the number of repetitions used for the periodic symmetry.

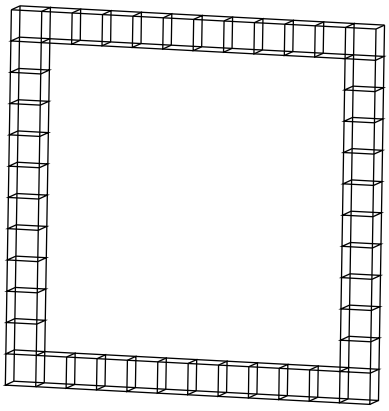


Figure 2: Three-dimensional square tube model.

Results and discussion

	Element 6, Side 5		
	RADFL	VFTOT	FTEMP
xrv38snp05.inp	849.4	0.6578	471.9
xrv38snp10.inp	2376.0	0.8696	495.4
xrv38snp20.inp	3722.0	0.9702	517.1
2D model	3823.0	1.0000	517.7

	Element 21, Side 4		
	RADFL	VFTOT	FTEMP
xrv38snp05.inp	−5592.0	0.7504	675.9
xrv38snp10.inp	−5525.0	0.8811	698.0
xrv38snp20.inp	−4648.0	0.9719	720.5
2D model	−4787.0	1.0000	719.5

Input files

xrv38snp05.inp Full cross-section model, DC3D8 elements, periodic symmetry (NR=5).
xrv38snp10.inp Full cross-section model, DC3D8 elements, periodic symmetry (NR=10).
xrv38snp20.inp Full cross-section model, DC3D8 elements, periodic symmetry (NR=20).

Infinitely long square section tube with blocking

Two-dimensional models

Elements tested

DC2D4

Problem description

Four different two-dimensional models of the cross-section of the square tube and the blocking object are used: the full model, a half model with one reflection symmetry, a quarter model with two reflection symmetries, and a quarter model with cyclic symmetry. The full, half, and quarter models are shown in *Figure 3*. The two-dimensional models imply that the tube and the blocking object extend infinitely in the direction normal to the cross-section.

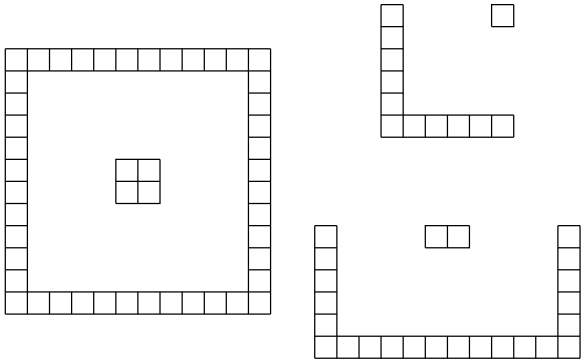


Figure 3: Two-dimensional square tubes with blocking.

Results and discussion

	Element 6, Side 3		
	RADFL	VFTOT	FTEMP
xrv24sb000.inp	1063.0	0.9970	701.0
xrv24sbr10.inp	1063.0	0.9970	701.0
xrv24sbr20.inp	1063.0	0.9970	701.0
xrv24sbc04.inp	1063.0	0.9970	701.0

	Element 21, Side 2		
	RADFL	VFTOT	FTEMP
xrv24sb000.inp	4506.0	0.9909	619.2
xrv24sbr10.inp	4506.0	0.9909	619.2
xrv24sbr20.inp	4506.0	0.9909	619.2
xrv24sbc04.inp	4506.0	0.9909	619.2

	Element 106, Side 1		
	RADFL	VFTOT	FTEMP
xrv24sb000.inp	−12745.0	1.0	812.6
xrv24sbr10.inp	−12745.0	1.0	812.6
xrv24sbr20.inp	−12745.0	1.0	812.6
xrv24sbc04.inp	−12745.0	1.0	812.6

Input files

xrv24sb000.inp	Full model, DC2D4 elements.
xrv24sbr10.inp	Half model, DC2D4 elements, one reflection symmetry.
xrv24sbr20.inp	Quarter model, DC2D4 elements, two reflection symmetries.
xrv24sbc04.inp	Quarter model, DC2D4 elements, cyclic symmetry (NC=4).

Three-dimensional models

Elements tested

DC3D8

Problem description

Six different models of the square section tube and the blocking object are used. These models involve different combinations of the cross-sectional model and the number of periodic symmetry repetitions used to simulate the infinite extent of the tube and the blocking object. Three cross-section models are used: the full model, a quarter model with two reflection symmetries, and a quarter model with cyclic symmetry. *Figure 4* shows the cross-section models used.

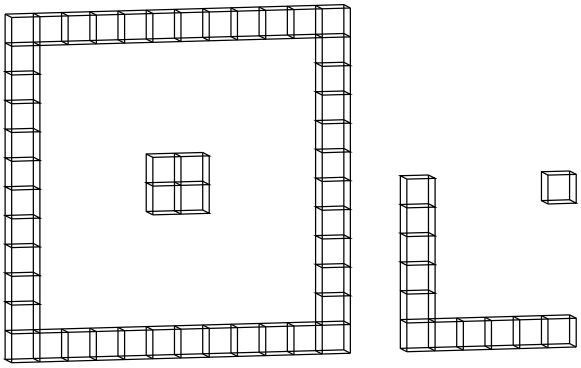


Figure 4: Three-dimensional square tubes with blocking.

Results and discussion

	Element 6, Side 5		
	RADFL	VFTOT	FTEMP
xrv38sbp05.inp	1044.0	0.7296	506.7
xrv38sbrp5.inp	1044.0	0.7296	506.7
xrv38sbcp5.inp	1044.0	0.7296	506.7
xrv38sbcp10.inp	1129.0	0.9016	637.9
xrv38sbcp20.inp	1098.0	0.9749	701.0
xrv38sbcp50.inp	1071.0	0.9934	700.7
2D model	1063.0	0.9970	701.0

	Element 21, Side 4		
	RADFL	VFTOT	FTEMP
xrv38sbp05.inp	1603.0	0.7893	511.4
xrv38sbrp5.inp	1603.0	0.7893	511.4
xrv38sbcp5.inp	1603.0	0.7893	511.4
xrv38sbcp10.inp	2930.0	0.9116	567.2
xrv38sbcp20.inp	4426.0	0.9710	618.3
xrv38sbcp50.inp	4484.0	0.9871	618.7

	Element 21, Side 4		
	RADFL	VFTOT	FTEMP
2D model	4506.0	0.9970	619.2

	Element 106, Side 3		
	RADFL	VFTOT	FTEMP
xrv38sbp05.inp	−15583.0	0.8510	776.8
xrv38sbrp5.inp	−15583.0	0.8510	776.8
xrv38sbcp5.inp	−15583.0	0.8510	776.8
xrv38sbcp10.inp	−14241.0	0.9593	790.6
xrv38sbcp20.inp	−12694.0	0.9875	813.6
xrv38sbcp50.inp	−12719.0	0.9926	812.9
2D model	−12745.0	1.0000	812.6

Input files

xrv38sbp05.inp	Full cross-section model, DC3D8 elements, periodic symmetry (NR=5).
xrv38sbrp5.inp	Quarter cross-section model with two reflection symmetries, DC3D8 elements, periodic symmetry (NR=5).
xrv38sbcp5.inp	Quarter cross-section model with cyclic symmetry (NC=4), DC3D8 elements, periodic symmetry (NR=5).
xrv38sbcp10.inp	Quarter cross-section model with cyclic symmetry (NC=4), DC3D8 elements, periodic symmetry (NR=10).
xrv38sbcp20.inp	Quarter cross-section model with cyclic symmetry (NC=4), DC3D8 elements, periodic symmetry (NR=20).
xrv38sbcp50.inp	Quarter cross-section model with cyclic symmetry (NC=4), DC3D8 elements, periodic symmetry (NR=50).

Finite length square section tube

Three-dimensional models without blocking

Elements tested

DC3D8

Problem description

A unit-length tube with a square cross-section is analyzed. Four different models of the square section are used: the full model, a half model with one reflection symmetry, a quarter model with two reflection symmetries, and a quarter model with cyclic symmetry. *Figure 5* shows the cross-section models used.

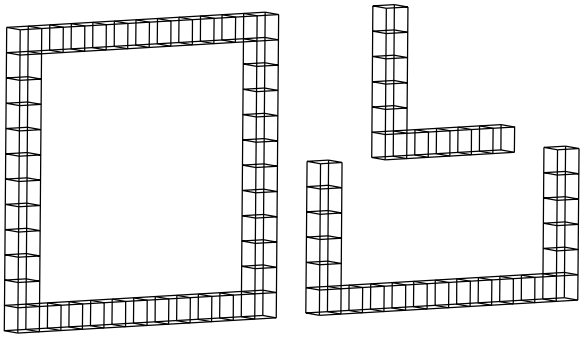


Figure 5: Three-dimensional finite square tubes with blocking.

Results and discussion

	Element 6, Side 5		
	RADFL	VFTOT	FTEMP
xrv38sn000.inp	−1544.0	0.0788	429.3
xrv38snr10.inp	−1544.0	0.0788	429.3
xrv38snr20.inp	−1544.0	0.0788	429.3
xrv38snc04.inp	−1544.0	0.0788	429.3

	Element 21, Side 4		
	RADFL	VFTOT	FTEMP
xrv38sn000.inp	−6694.0	0.2796	629.1
xrv38snr10.inp	−6694.0	0.2796	629.1
xrv38snr20.inp	−6694.0	0.2796	629.1
xrv38snc04.inp	−6694.0	0.2796	629.1

Input files

xrv38sn000.inp	Full cross-section model, DC3D8 elements.
xrv38snr10.inp	Half cross-section model, DC3D8 elements, one reflection symmetry.
xrv38snr20.inp	Quarter cross-section model, DC3D8 elements, two reflection symmetries.
xrv38snc04.inp	Quarter cross-section model, DC3D8 elements, cyclic symmetry (NC=4).

Three-dimensional models with blocking

Elements tested

DC3D8

Problem description

A unit-length square cross-section tube and a blocking object are analyzed. Three cross-section models are used: the full model, a quarter model with two reflection symmetries, and a quarter model with cyclic symmetry. *Figure 6* shows the cross-section models used.

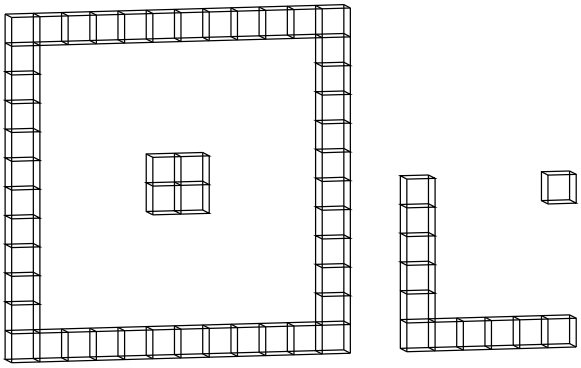


Figure 6: Three-dimensional finite square tubes with blocking.

Results and discussion

	Element 6, Side 5		
	RADFL	VFTOT	FTEMP
xrv38sb000.inp	−169.5	0.1000	359.6
xrv38sbr20.inp	−169.5	0.1000	359.6
xrv38sbc04.inp	−169.5	0.1000	359.6

	Element 21, Side 4		
	RADFL	VFTOT	FTEMP
xrv38sb000.inp	−452.7	0.2874	372.1
xrv38sbr20.inp	−452.7	0.2874	372.1
xrv38sbc04.inp	−452.7	0.2874	372.1

	Element 106, Side 3		
	RADFL	VFTOT	FTEMP
xrv38sb000.inp	−17304.0	0.1322	745.5
xrv38sbr20.inp	−17304.0	0.1322	745.5
xrv38sbc04.inp	−17304.0	0.1322	745.5

Input files

xrv38sb000.inp	Full cross-section model, DC3D8 elements.
xrv38sbr20.inp	Quarter cross-section model, DC3D8 elements, two reflection symmetries.
xrv38sbc04.inp	Quarter cross-section model, DC3D8 elements, cyclic symmetry (NC=4).

Square section tubular ring

Axisymmetric models without blocking

Elements tested

DCAX4

Problem description

A tubular ring with a square cross-section is analyzed. Two different models of the square section are used: the full model and a half model with one reflection symmetry. *Figure 7* shows the cross-section models used.

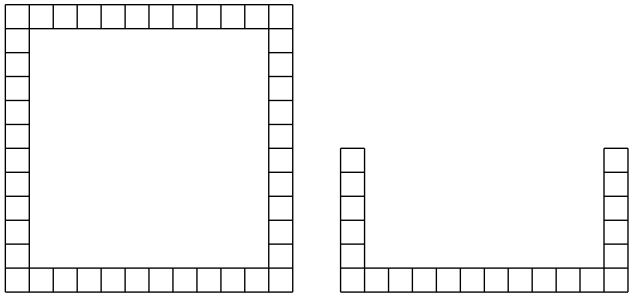


Figure 7: Axisymmetric models without blocking.

Results and discussion

	Element 6, Side 3		
	RADFL	**VFTOT**	**FTEMP**
xrva4sn000.inp	−218.7	1.003	563.3
xrva4snr10.inp	−218.7	1.003	563.3

	Element 21, Side 2		
	RADFL	**VFTOT**	**FTEMP**
xrva4sn000.inp	−5599.0	1.022	690.5
xrva4snr10.inp	−5599.0	1.022	690.5

Input files

xrva4sn000.inp	Full cross-section model, DCAX4 elements.
xrva4snr10.inp	Half cross-section model, DCAX4 elements, one reflection symmetry.

Axisymmetric models with blocking

Elements tested

DCAX4

Problem description

A square cross-section tubular ring with a blocking object inside it is analyzed. Two different models of the square section are used: the full model and a half model with one reflection symmetry. *Figure 8* shows the cross-section models used.

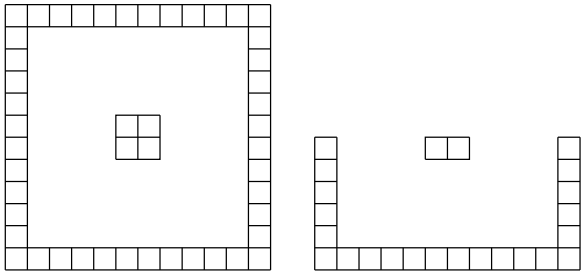


Figure 8: Axisymmetric models with blocking.

Results and discussion

	Element 6, Side 3		
	RADFL	**VFTOT**	**FTEMP**
xrva4sb000.inp	1225.0	1.004	711.3
xrva4sbr10.inp	1225.0	1.004	711.3

	Element 21, Side 2		
	RADFL	**VFTOT**	**FTEMP**
xrva4sb000.inp	3970.0	1.015	641.0
xrva4sbr10.inp	3970.0	1.015	641.0

	Element 106, Side 1		
	RADFL	**VFTOT**	**FTEMP**
xrva4sb000.inp	−12877.0	1.003	817.5
xrva4sbr10.inp	−12877.0	1.003	817.5

Input files

xrva4sb000.inp	Full cross-section model, DCAX4 elements.
xrva4sbr10.inp	Half cross-section model, DCAX4 elements, one reflection symmetry.

Infinitely extending three-dimensional array of cubic objects

Two-dimensional models

Elements tested

DC2D4

Problem description

An infinite array of cubic objects is simulated. The two-dimensional models imply that the array extends to infinity in the third direction. Three different models are used: an array of nine by eleven objects, an array of nine objects with periodic symmetry in the direction perpendicular to the array, and a single object with periodic symmetry in two directions. The number of repetitions in the models using periodic symmetry makes these models equivalent to the nine by eleven array model. The models are shown in *Figure 9* where the black square

represents the model with two periodic symmetries and the gray squares represent the model with one periodic symmetry.

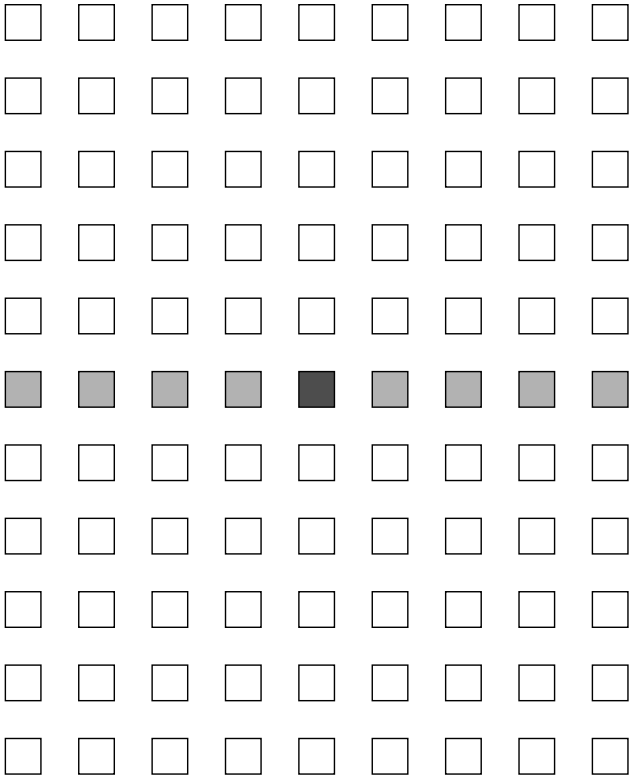


Figure 9: Two-dimensional cubic array.

Results and discussion

	Element 55, Side 1		
	RADFL	**VFTOT**	**FTEMP**
xrv24ab000.inp	−24725.0	0.9635	884.6
xrv24abp05.inp	−24723.0	0.9635	884.6
xrv24ab2p5.inp	−23444.0	0.9635	887.4

	Element 55, Side 2		
	RADFL	**VFTOT**	**FTEMP**
xrv24ab000.inp	23168.0	0.9645	594.8
xrv24abp05.inp	23175.0	0.9645	594.9
xrv24ab2p5.inp	23465.0	0.9645	603.8

Input files

xrv24ab000.inp — Nine by eleven array, DC2D4 elements.

xrv24abp05.inp — Nine object array with one periodic symmetry (NR=5), DC2D4 elements.

xrv24ab2p5.inp — Single object array with two periodic symmetries (NR_1=4, NR_2=5), DC2D4 elements.

Three-dimensional models

Elements tested

DC3D8

Problem description

An infinite array of cubic objects is simulated. The three-dimensional models consist of a single cubic element with periodic symmetry in three directions. Two models are used where the number of periodic symmetry repetitions is varied. The single element on which the models are based is shown in *Figure 10*.

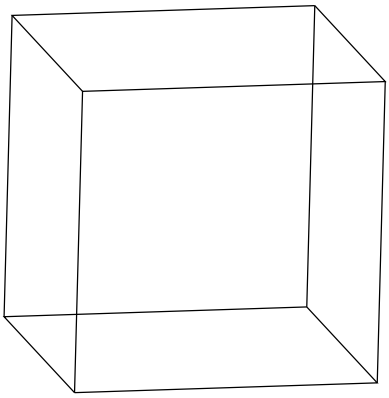


Figure 10: Single element used for three-dimensional cubic array.

Results and discussion

	Element 55, Side 3		
	RADFL	**VFTOT**	**FTEMP**
xrv38abp05.inp	−6657.0	1.011	805.8
xrv38abp10.inp	−7044.0	1.086	806.1

	Element 55, Side 4		
	RADFL	**VFTOT**	**FTEMP**
xrv38abp05.inp	6527.0	1.012	722.2
xrv38abp10.inp	7026.0	1.083	723.8

Input files

xrv38abp05.inp — Single object array with three periodic symmetries (NR_1=4, NR_2=4, NR_3=5), DC3D8 elements.

xrv38abp10.inp — Single object array with three periodic symmetries (NR_1=8, NR_2=8, NR_3=10), DC3D8 elements.

Infinitely long finned tube inside another infinitely long tube

Axisymmetric models

Elements tested

DCAX4

Problem description

Radiation between an infinitely long, finned tube inside another infinitely long simple tube is simulated. The axisymmetric mesh used is shown in *Figure 11*. The infinite extent of the tubes is modeled with periodic symmetry in the direction of the length of the tubes. Three models with a varying number of repetitions for the periodic symmetry are used.

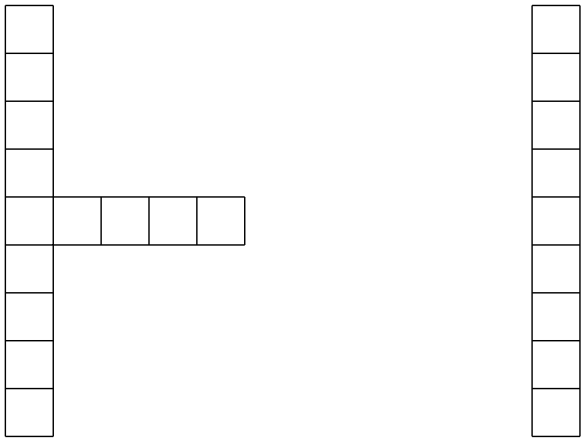


Figure 11: Axisymmetric mesh for finned tube models.

Results and discussion

	Element 82, Side 1		
	RADFL	VFTOT	FTEMP
xrva4tb000.inp	−6118.0	0.4383	710.0
xrva4tbp05.inp	−5885.0	1.022	747.0
xrva4tbp10.inp	−5884.0	1.022	747.0

	Element 85, Side 3		
	RADFL	VFTOT	FTEMP
xrva4tb000.inp	−755.8	0.2182	487.8
xrva4tbp05.inp	524.6	0.9951	589.1
xrva4tbp10.inp	508.9	1.002	588.8

	Element 92, Side 4		
	RADFL	VFTOT	FTEMP
xrva4tb000.inp	1875.0	0.3750	415.7
xrva4tbp05.inp	6404.0	1.005	459.1

	Element 92, Side 4		
	RADFL	VFTOT	FTEMP
xrva4tbp10.inp	6404.0	1.012	459.1

Input files

xrva4tb000.inp	Axisymmetric model without periodic symmetry, DCAX4 elements.
xrva4tbp05.inp	Axisymmetric model with periodic symmetry (NR=5), DCAX4 elements.
xrva4tbp10.inp	Axisymmetric model with periodic symmetry (NR=10), DCAX4 elements.

Three-dimensional models

Elements tested

DC3D8

Problem description

Radiation between an infinitely long finned tube inside another infinitely long simple tube is simulated. The two three-dimensional meshes used are shown in *Figure 12*: one is a full 360° mesh, and the other is a slice of this mesh that is used in conjunction with cyclic symmetry. The number of cycles used in the cyclic symmetry is varied. The infinite extent of the tubes is modeled with periodic symmetry in the direction of the length of the tubes.

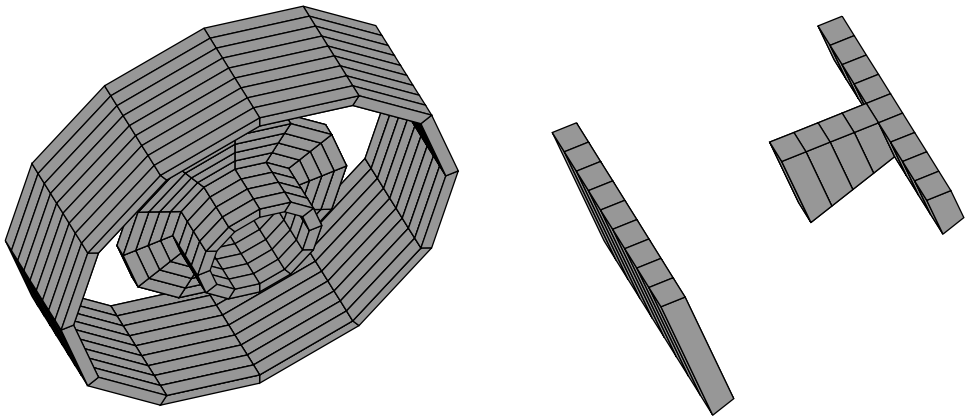


Figure 12: Three-dimensional meshes for finned tube models.

Results and discussion

	Element 82, Side 3		
	RADFL	VFTOT	FTEMP
xrv38tb000.inp	−6217.0	0.4003	692.1
axisymmetric model	−6118.0	0.4383	710.0
xrv38tbp05.inp	−5721.0	0.9710	735.8
xrv38tbpc12.inp	−6362.0	0.9710	750.6
xrv38tbpc24.inp	−6070.0	1.010	747.2
axisymmetric model	−5885.0	1.022	747.0

	Element 85, Side 5		
	RADFL	VFTOT	FTEMP
xrv38tb000.inp	−722.4	0.2256	497.9
axisymmetric model	−755.8	0.2182	487.8
xrv38tbp05.inp	424.3	0.9987	587.4
xrv38tbpc12.inp	439.2	0.9987	591.6
xrv38tbpc24.inp	507.6	0.9964	589.2
axisymmetric model	524.6	0.9951	589.1

	Element 92, Side 6		
	RADFL	VFTOT	FTEMP
xrv38tb000.inp	1791.0	0.3787	414.7
axisymmetric model	1875.0	0.3750	415.7
xrv38tbp05.inp	6219.0	0.9844	455.5
xrv38tbpc12.inp	6465.0	0.9744	457.7
xrv38tbpc24.inp	6438.0	1.027	458.9
axisymmetric model	6404.0	1.012	459.1

Input files

xrv38tb000.inp — Full 360° model without periodic symmetry in the infinite direction, DC3D8 elements.

xrv38tbp05.inp — Full 360° model with periodic symmetry in the infinite direction (NR=5), DC3D8 elements.

xrv38tbpc12.inp — 30° slice model with cyclic symmetry (NC=12) and periodic symmetry in the infinite direction (NR=5), DC3D8 elements.

xrv38tbpc24.inp — 15° slice model with cyclic symmetry (NC=24) and periodic symmetry in the infinite direction (NR=5), DC3D8 elements.

Release rotational degrees of freedom

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

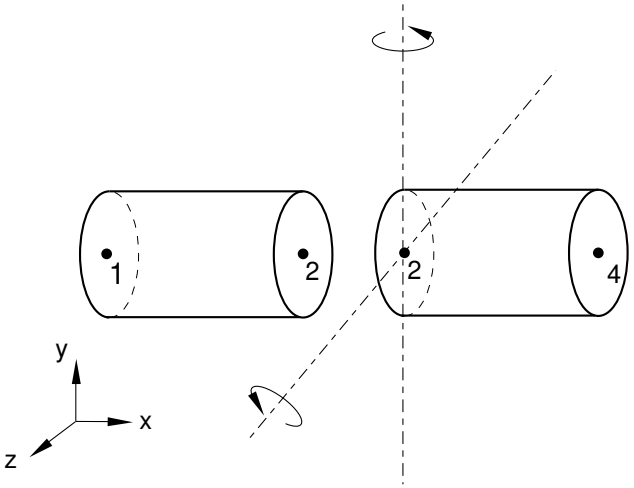
Elements tested

B21 B22 B23 B31 B32 B33

Features tested

Various types of hinged connections are tested by releasing one or more rotational degrees of freedom. Equivalent models using multi-point constraints are included for comparison.

Problem description



Two beam elements are aligned with the *x*-axis, joined at the center, and clamped at nodes 1 and 4. Rotational degrees of freedom are released at the center, node 2. Equivalent MPC definitions are used to connect two separate nodes at the center, nodes 2 and 3.

Loading: Step 1: The left half of the model is loaded by forces per unit length, PY = −1000 and PZ = 1000.

The right half of the model is loaded by forces per unit length, PY = 1000 and PZ = −1000.

Step 2: The loads that were applied in the first step are applied again, this time using NLGEOM for large-displacement analysis.

Results and discussion

The results are the same for the equivalent multi-point constraint models.

Input files

xreleasepinx2.inp	Static steps, release all the rotational degrees of freedom for a beam in a plane.
xmpcpinx2.inp	Static steps, equivalent MPC type PIN for a beam in a plane.
xreleasepinx3.inp	Static steps, release all the rotational degrees of freedom for a beam in space.
xmpcpinx3.inp	Static steps, equivalent MPC type PIN for a beam in space.
xreleaserevo2.inp	Static steps, release one rotational degree of freedom for a beam in space.
xmpcrevo2.inp	Static steps, equivalent MPC type REVOLUTE plus MPC type PIN for a beam in space.
xreleaseuniv2.inp	Static steps, release two rotational degrees of freedom for a beam in space.
xmpcuniv2.inp	Static steps, equivalent MPC type UNIVERSAL plus MPC type PIN for a beam in space.
xreleaseuniv3.inp	Frequency, steady-state dynamics, modal dynamic, and response spectrum steps; release two rotational degrees of freedom for a beam in space.
xmpcuniv3.inp	Frequency, steady-state dynamics, modal dynamic, and response spectrum steps; equivalent MPC type UNIVERSAL plus MPC type PIN for a beam in space.

Extrapolation

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

B21 B31 CPS4 M3D4 S4R T2D2

Features tested

Linear extrapolation, parabolic extrapolation, and suppression of extrapolation are tested.

Problem description

These tests verify the performance of extrapolation for structural and continuum elements used in models subjected to an in-plane rotation of 45°. For elements that do not have rotation degrees of freedom, beam elements are used to connect the elements to the point of rotation. The restart test verifies that the solution history information required for the extrapolation algorithm is transferred correctly to a restarted analysis.

Results and discussion

In all cases the results indicate that this option performs as expected. When parabolic extrapolation is used, there is a speedup in computational time compared to linear extrapolation. The restart analysis results are identical to those for the original analysis from which the restart was run.

Input files

xstebeam.inp	Beam (B31) elements with parabolic extrapolation.
xstecon.inp	Continuum (CPS4) and beam (B31) elements with linear extrapolation.
xstememb.inp	Membrane (M3D4) and beam (B31) elements with parabolic extrapolation.
xsteshel.inp	Shell (S4R) element with parabolic extrapolation.
xsteshre.inp	Restart of xsteshel.inp without terminating the current step in the analysis from which the restart is made.
xstetrs.inp	Truss (T2D2) and beam (B21) elements with parabolic extrapolation.

Surface-based fluid cavities

Products: Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Features tested

This section provides basic verification tests to define:

- Fluid behavior for a surface-based cavity
- Fluid exchange
- Fluid exchange activation
- Fluid exchange in or out of a fluid cavity
- Fluid inflator
- Fluid inflator activation
- Fluid inflator property
- Gas species used for a fluid inflator
- Molar heat capacity
- Molecular weight
- Surface-based fluid cavity

Fluid behavior

Problem description

In this test the following three types of fluid behaviors are tested:

- Fluid cavity filled with a mixture of gases (pneumatic fluids) under isothermal conditions.
- Fluid cavity filled with a mixture of gases (pneumatic fluids) under adiabatic conditions with optional temperature dependence of heat capacity.
- Fluid cavity filled with an hydraulic fluid with optional temperature dependence of fluid density.

Five independent fluid cavities (no fluid exchange) are modeled using the surface-based fluid cavity capability, each with a different fluid behavior.

Results and discussion

The test verifies that Abaqus/Explicit accurately addresses the relationship between fluid pressure, fluid temperature, and fluid volume. In addition, the test also verifies the use of the magnitude of the additional and minimum volumes for the fluid.

Input files

fluidbehavior.inp	Tests the behavior of pneumatic and hydraulic fluids.

Gas structure interaction

Elements tested

B21
CAX3 CAX4R C3D4 C3D6 C3D8R C3D10M CPE3 CPE4R CPS3 CPS4R
M3D3 M3D4R
RAX2 R2D2 R3D3 R3D4
S3R S4R SAX1 SC6R SC8R
SFM3D3 SFM3D4R
T2D2

Problem description

A fluid cavity is primarily defined to consider the coupling between the deformation of the structure and the pressure exerted by the fluid on the structure. These tests verify the capability of Abaqus/Explicit to model this interdependence accurately by defining a fluid cavity based on the surfaces of the structure. The structure enclosing the fluid cavity is modeled using different feasible combinations of finite elements. The volume of the cavity is changed intentionally during the analysis by prescribing displacement boundary conditions on a particular set of nodes, which results in a change in the cavity pressure.

Results and discussion

The results indicate that the change in cavity pressure gets correctly transferred to the elements of the structure and is reflected as a change in the nodal reaction forces.

Input files

gasstructure_3d.inp	The structure enclosing the fluid cavity is modeled using different three-dimensional finite elements available in Abaqus/Explicit.
gasstructure_c3d10m.inp	The structure enclosing the fluid cavity is modeled using C3D10M elements.
gasstructure_2d.inp	The structure enclosing the fluid cavity is modeled using different two-dimensional finite elements available in Abaqus/Explicit.
gasstructure_axi.inp	The structure enclosing the fluid cavity is modeled using different axisymmetric finite elements available in Abaqus/Explicit.

Fluid exchange

Problem description

In this test fluid flow between a cavity and its environment or between two fluid cavities is modeled. Test cases include flow of a single gas, flow of a mixture of gases, and flow of hydraulic fluids. For pneumatic fluids, both isothermal and adiabatic behaviors are tested.

Results and discussion

The analysis results closely match with the analytical results, which are obtained using the governing equations described in *Fluid Exchange Definition*.

Input files

fluidexchange_pneumatic.inp	Flow between a single cavity and its environment and between two fluid cavities filled with either a single gas (pneumatic fluid) or a mixture of gases (pneumatic fluids) modeled using all fluid exchange property options.
fluidexchange_hydraulic.inp	Flow between a single cavity and its environment and between two fluid cavities filled with an hydraulic fluid modeled using all fluid exchange property options.
fluidexchange_usereffarea.inp	Flow between a single cavity and its environment with leakage area defined using user subroutine *VUFLUIDEXCHEFFAREA*.
vufluidexcheffarea.f	User subroutine used in `fluidexchange_usereffarea.inp`.
fluidexchange_cavitypres.inp	Application of fluid pressure on the fluid exchange surface.

Fluid inflators

Problem description

This test verifies the fluid inflator properties that can be defined in Abaqus/Explicit to simulate the flow characteristics of the actual inflators. The inflator mass flow rate and inflator temperature are assumed to be linearly varying with time for the mass flow rate and inflator gas temperature type of fluid inflator property. For the tank volume type of inflator property, the tank volume and tank pressure are set to be the same as the cavity volume and cavity pressure obtained in the mass flow rate and inflator gas temperature case. For the dual pressure type of fluid inflator property definition, the tank volume and tank pressure data are taken from the tank volume case and the inflator pressures at different inflation times are determined from the equations given in *Inflator Definition*. The data necessary to define the mass flow rate and inflator pressure type of inflator property are obtained from the previous three cases. In the test a total of ten fluid cavities are modeled using the surface-based fluid cavity capability. Fluid cavities 1–8 and 10 are inflated with the same ideal gas or a mixture of ideal gases that are initially present in the cavity. However, the molar mass fractions of the gases inflating the fluid cavity filled with a mixture are considered to be different from the initial molar mass fractions. In the case of cavity 9, the constituents of the gas mixture inflating the cavity are considered to be different from the constituents present in the cavity initially.

Results and discussion

The results for the mass flow rate and gas inflator temperature type of fluid inflator property are in agreement with the analytical results. The results for both the tank volume and mass flow rate and inflator pressure type of inflator properties, as expected, are almost the same as for the mass flow rate and gas inflator temperature type of inflator property. However, for the dual pressure type of inflator property, the results do not match the results of the previous cases since the heat capacity for the ideal gases is considered to be dependent on temperature.

Input files

fluidinflators.inp	Tests fluid inflator properties.

Defining temperature, field variable, pressure stress, and pore fluid pressure values

Products: Abaqus/Standard Abaqus/Explicit

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Features tested

Applications of the temperature, field variable, pressure stress, and pore fluid pressure procedures are tested. The first set of tests verifies that temperature and field variable data are properly transferred from a heat transfer analysis to a structural analysis. The second set of tests verifies the use of these commands in conjunction with composite structural shells. The third set of tests verifies the interpolation of temperatures to the midside nodes in a sequential thermal-stress analysis, when the heat transfer analysis is carried out using first-order elements and the stress analysis is carried out using second-order elements. The fourth set of tests verifies that temperatures are properly interpolated between dissimilar meshes. Heat transfer models and stress analysis models may have dissimilar meshes, and the nodal temperatures for the current model will be interpolated from the nodal temperatures from the heat transfer model. The fifth set of tests verifies that temperatures and pressures are properly defined using data line input for various combinations of these two commands. The sixth set of tests verifies that a solution-dependent variable from a heat transfer analysis is properly transferred as a field variable into a stress analysis. The seventh set of tests verifies that a pore fluid pressure field is properly transferred as a predefined field variable into a stress analysis.

In several of the tests zero-increment results file output is requested. This output is used to define initial values of temperature, field variables, and pressure stress for subsequent structural analyses.

Reading temperature and field variable data from results files

Elements tested

DC1D2 T3D2

Problem description

These tests verify that temperature and field variable values are properly transferred to a structure. The structure being analyzed is a cantilevered truss made up of 10 T3D2 elements.

Three different transient heat transfer runs are used to generate three results files containing temperature histories. These files will be read into subsequent stress analyses as either temperature or field variable data. All of the runs begin with the entire truss at some initial temperature; the temperature throughout the truss is then ramped to some new temperature.

The three heat transfer runs are as follows:

xtfvtrt1.inp

Initial temperature: 100

Final temperature: 200

xtfvtrt2.inp

Initial temperature: 200

Final temperature: 250

xtfvtrt3.inp

Initial temperature: 200

Final temperature, Step 1: 180

Final temperature, Step 2: 100

The subsequent stress analysis runs are as follows:

xtfvtrs1.inp

This file tests the setting of temperature and more than one field variable using results files. Temperature and two field variables are set by reading the data from the results files of the heat transfer runs as follows:

xtfvtrt1.fil ⇒ Temperature

xtfvtrt2.fil ⇒ Field variable 1

xtfvtrt1.fil ⇒ Field variable 2

xtfvtrs2.inp

This file tests the setting of a field variable from a results file without temperature being present in the problem. This test is important because of the way that temperatures and field variables are stored internally. The field variable is set by reading the data from the results file of the first heat transfer run as follows:

xtfvtrt1.fil ⇒ Field variable

xtfvtrs3.inp

This file tests the presence of temperatures and field variables when initial condition specifications are present for variables that are not used in the analysis. Initial conditions are given for temperature and two field variables, and then only temperature and the first field variable are set by results files. In addition, two procedures are included for the same field variable to test that only the last command is used. Temperature and the field variable are set by reading the data from the results files of the heat transfer runs as follows:

xtfvtrt1.fil ⇒ Temperature

xtfvtrt2.fil ⇒ Field variable 1

xtfvtrs4.inp

This is a three-step problem involving temperature and one field variable. In the first step an amplitude curve is used to set the temperature to 200 and the field variable to 250. In the second step the temperature is ramped down to 150, and the field variable is defined by the results file from xtfvtrt2.fil. In the third step both the temperature and the field variable are reset to their initial conditions.

The following must be confirmed by this test:

- Temperatures and field variables must be set correctly using an amplitude curve.

- Initial conditions must be ignored if temperatures and field variables are set using an amplitude curve.
- Results file data must be scaled properly in time if the stress analysis time period is different from the heat analysis time period.
- If commands are given to read temperature/field variable data both from data lines and from a results file, the data line input must take precedence.
- If the OP parameter is given with a value of NEW, temperatures/field variables must be ramped back to initial conditions or set to the new values defined on the data lines.

xtfvtrsr.inp

This analysis restarts xtfvtrs4.inp from the third step. Two additional steps are performed. In the first step the temperature is set by reading the results file from xtfvtrt1.fil, and the field variable is set by reading the results file from xtfvtrt2.fil. In the second step the temperature is set using the data from the second step of the results file from xtfvtrt3.fil.

xtfvtrs5.inp

This run sets the beginning and end step and increment numbers for the history data in temperature and field values. Temperature and two field variables are set by reading the data from the results files of the heat transfer runs as follows:

xtfvtrt1.fil ⇒ Temperature, Beginning increment=1, End increment=5

xtfvtrt2.fil ⇒ Field variable 1, Beginning increment=5, End increment=8

xtfvtrt1.fil ⇒ Field variable 2, Beginning increment=6, End increment=10

All data are read from Step 1, so the beginning and end step are both 1 in all cases.

Results and discussion

The exact solution to the heat transfer problems (*xtfvtrt1.inp*, *xtfvtrt2.inp*) consists of a linear temperature history. Temperature is uniform throughout the structure at each point in time. The solution given by Abaqus matches the exact solution.

The only quantity of interest in the stress analysis runs is the temperature in the structure. Expected solutions are shown in *Figure 1* through *Figure 5*.

Input files

xtfvtrt1.inp	Truss, heat transfer, first run.
xtfvtrt2.inp	Truss, heat transfer, second run.
xtfvtrt3.inp	Truss, heat transfer, third run.
xtfvtrs1.inp	Truss, stress analysis, first run.
xtfvtrs2.inp	Truss, stress analysis, second run.
xtfvtrs3.inp	Truss, stress analysis, third run.
xtfvtrs4.inp	Truss, stress analysis, fourth run.
xtfvtrsr.inp	Truss, stress analysis, restart.
xtfvtrs5.inp	Truss, stress analysis, fifth run.

Figures

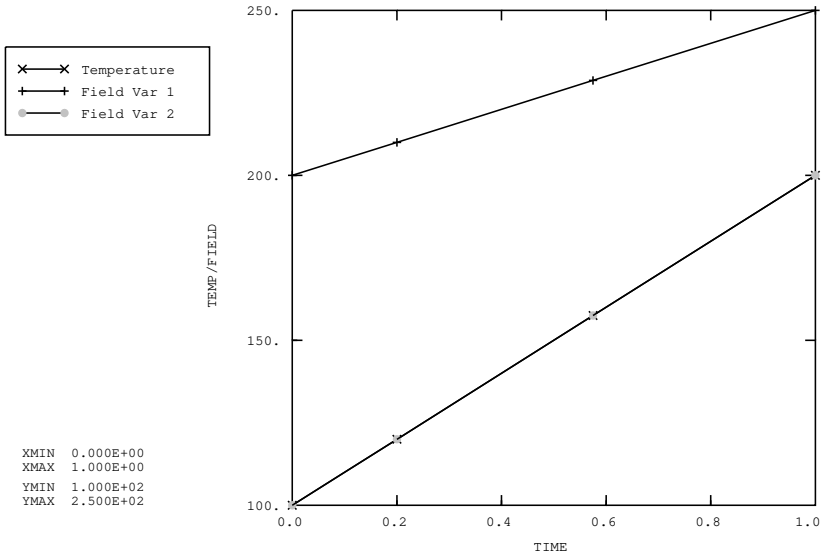


Figure 1: Temperature and field variables for xtfvtrs1.inp.

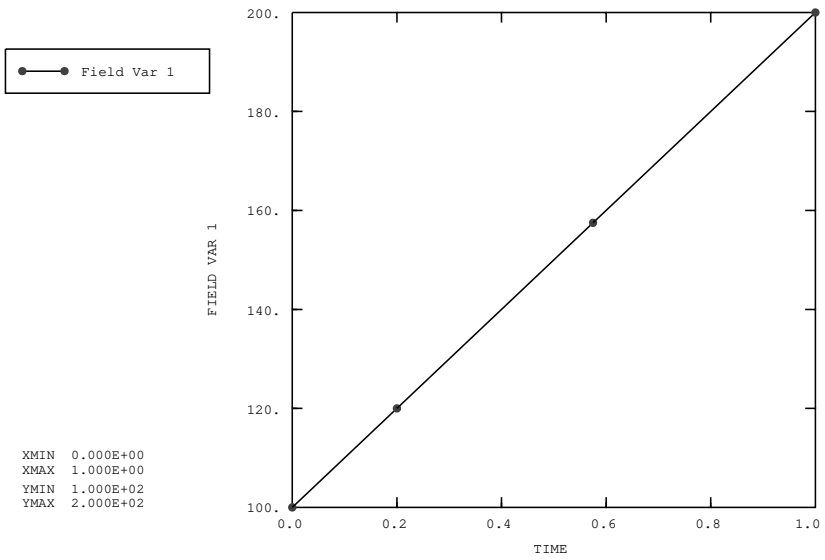


Figure 2: Field variable for xtfvtrs2.inp.

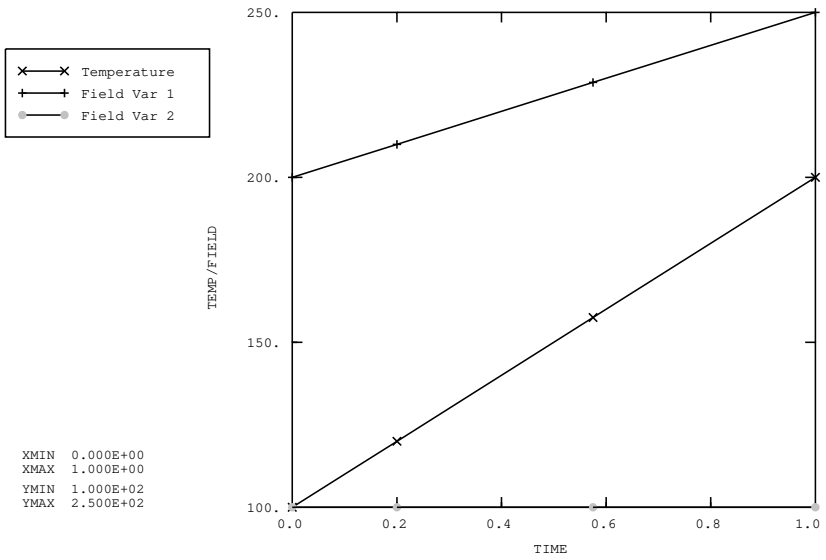


Figure 3: Temperature and field variable for xtfvtrs3.inp.

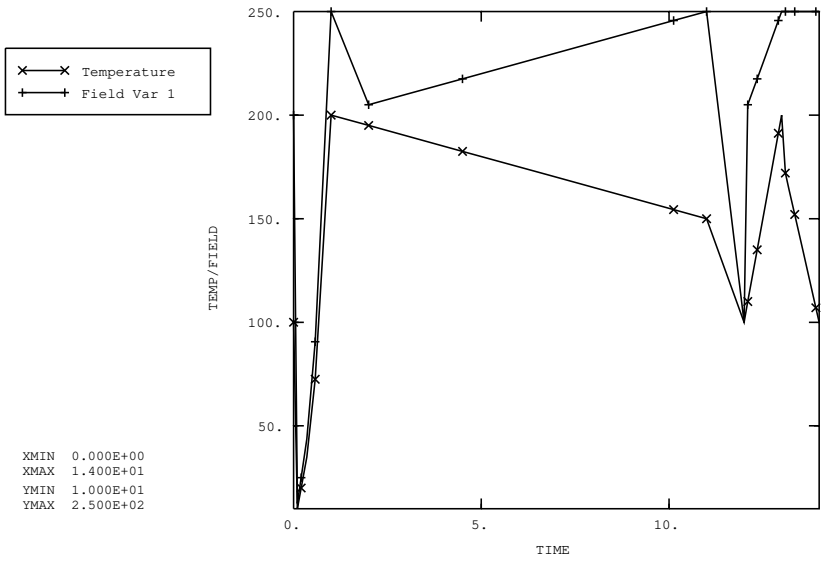


Figure 4: Temperature and field variable for xtfvtrs4.in and xtfvtrsr.inp.

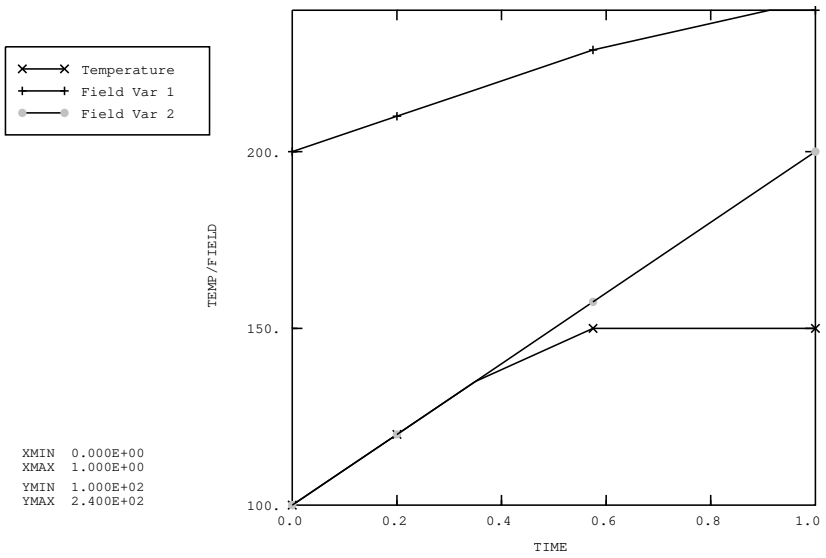


Figure 5: Temperature and field variables for xtfvtrs5.inp.

Composite shell temperature loads

Problem description

In Abaqus/Standard these tests verify the use of predefined temperature and field variables in conjunction with composite structural shells. Both temperature and field variable results are generated from a single previously run heat transfer shell analysis. The same analysis can be used to generate field variable results since field variables are stored identically to temperatures in an Abaqus results file.

A steady-state heat transfer analysis is performed to obtain the temperature distribution through the thickness of the composite layers. The heat transfer problem involves a three-layer composite shell that is subjected to prescribed thermal boundary conditions on its top and bottom surfaces.

In the subsequent structural shell models, five section points per layer are used. The temperatures and field variables are assigned to these five points through a linear interpolation of the three values available per layer from the preceding heat transfer analysis. The results of these analyses verify that the temperatures and field variables are assigned properly.

In Abaqus/Explicit, instead of a sequential analysis, a transient coupled dynamic temperature-displacement analysis is performed on a three-layer composite shell that is subjected to prescribed thermal boundary conditions on its top and bottom surfaces. A sufficiently large step time is prescribed such that the analysis can reach the steady-state regime. Three temperature points are used for each layer. The temperature distribution obtained is compared to the exact solution.

Results and discussion

The heat transfer run matches the exact solution for the temperature distribution through the composite shell layers. In addition, these values are transferred properly in Abaqus/Standard to the structural composite shell as either temperature or a field variable. In Abaqus/Explicit both heat-transfer and stress analyses are solved simultaneously, and the results match the analytical solution and the Abaqus/Standard solution.

- The temperature/field variable at the bottom of layer 1 is 425°.
- The temperature/field variable at the top of layer 1 and the bottom of layer 2 is 373.2°.

- The temperature/field variable at the top of layer 2 and the bottom of layer 3 is 336.8°.
- The temperature/field variable at the top of layer 3 is 287.5°.

There is a linear variation of temperature or field variable between the top and bottom of each layer.

Input files

xtmpcst3.inp	Heat transfer analysis; DS3 elements.
xtmpcss3.inp	Stress analysis; temperature results; DS3 elements.
xfvcss3x.inp	Stress analysis; field variable results; DS3 elements.
xtmpcst4.inp	Heat transfer analysis; DS4 elements.
xtmpcst4.f	User subroutine *FILM* used in xtmpcst4.inp.
xtmpcss4.inp	Stress analysis; temperature results; DS4 elements.
xfvcss4x.inp	Stress analysis; field variable results; DS4 elements.
xtmpcst6.inp	Heat transfer analysis; DS6 elements.
xtmpcss6.inp	Stress analysis; temperature results; DS6 elements.
xfvcss6x.inp	Stress analysis; field variable results; DS6 elements.
xtmpcst8.inp	Heat transfer analysis; DS8 elements.
xtmpcss8.inp	Stress analysis; temperature results; DS8 elements.
xfvcss8x.inp	Stress analysis; field variable results; DS8 elements.
compshell_tempload_s4rt_xpl.inp	Dynamic temperature-displacement analysis; Abaqus/Explicit; S4RT elements.
compshell_tempload_s3rt_xpl.inp	Dynamic temperature-displacement analysis; Abaqus/Explicit; S3RT elements.

Temperature interpolation to midside nodes

Problem description

These tests verify the interpolation of temperatures to the midside nodes of higher-order elements in a sequential thermal-stress analysis, when the heat transfer analysis is performed using first-order elements and the stress analysis is carried out using second-order elements.

The results of the heat transfer analyses are read into the stress analyses, and the initial conditions applied to the heat transfer analysis are read into the stress analyses. In both analyses the temperatures at the midside nodes are interpolated from the corner nodes of the element. Temperature interpolation is carried out on an edgewise basis for each element. Thus, the temperature at the midside node of an element is interpolated linearly from the temperatures at the corresponding corner nodes.

The midside node temperature interpolation is tested for one-dimensional, two-dimensional, and three-dimensional elements.

Only one element is used in the finite element models for both heat transfer analysis and stress analysis. Arbitrary material properties are assumed.

Results and discussion

The results of the stress analysis with higher-order elements compare well with those obtained with linear elements.

Input files

Heat transfer analyses:

xtdc1d2h.inp	DC1D2 elements.
xtdc2d3h.inp	DC2D3 elements.
xtdc2d4h.inp	DC2D4 elements.
xtdc3d4h.inp	DC3D4 elements.
xtdc3d6h.inp	DC3D6 elements.
xtdc3d8h.inp	DC3D8 elements.

Stress analyses using Abaqus/Standard:

xtc3d10s.inp	C3D10 elements.
xtc3d10ms.inp	C3D10M elements.
xtc3d15s.inp	C3D15 elements.
xtc3d20s.inp	C3D20 elements.
xtcpe6ss.inp	CPE6 elements.
xtcpe6ms.inp	CPE6M elements.
xtcpe8ss.inp	CPE8 elements.
xtt2d3ss.inp	T2D3 elements.

Stress analyses using Abaqus/Explicit:

xtc3d10ms_xpl.inp	C3D10M elements.
xtcpe6ms_xpl.inp	CPE6M elements.

The input files for the stress analyses with linear elements can be generated by suitably replacing the element type in the above files.

Temperature interpolation between dissimilar meshes

Problem description

These tests verify the interpolation of temperatures between dissimilar meshes. This capability is available only for use with the output database file. The temperatures must be interpolated from the nodes of the heat transfer models to the nodes of the current stress analysis models. For the cases where the only dissimilarity is an element order, the temperatures at the midside nodes should be interpolated from the corner nodes of the element. However, for the purpose of verification we reused some of the models created for the midside cases.

The results of the heat transfer (or coupled temperature-displacement) analyses are read into the stress analyses. The temperatures must be interpolated from the nodes of the element in the heat transfer models to the nodes of the current stress analysis models.

The interpolation technique is tested for two-dimensional and three-dimensional elements.

Results and discussion

The temperature distribution in the stress analysis models compares well with that obtained in the heat transfer (coupled temperature-displacements) models.

Input files

Coupled temperature-displacement and stress analyses:

pgc38ths.inp	C3D8HT elements. NT field written to the output database.
psc38ths_inter.inp	C3D8H elements. Static analysis with the temperature field interpolated from psc38ths_inter.inp.

Heat transfer analyses:

xtdc2d3h.inp	DC2D3 elements.
xtdc2d4h.inp	DC2D4 elements.
xtdc3d4h.inp	DC3D4 elements.
xtdc3d6h.inp	DC3D6 elements.
pgce4shm_inter.inp	DC2D4 elements.
SoFoTiedMixMesh.inp	DC3D8, DC3D10 elements.
xtcds4.inp	DS4 elements.

Stress analyses using Abaqus/Standard:

xtc3d10s_interpolate.inp	C3D10 elements.
xtc3d10ms_interpolate.inp	C3D10M elements.
xtc3d15s_interpolate.inp	C3D15 elements.
xtcpe6ms_interpolate.inp	CPE6M elements.
sub_driven.inp	C3D8 elements.
xtcpe8ss_interpolate.inp	CPE8 elements.
psce4sh1_inter.inp	CPE4 elements.
psce4sh1_inter_res.inp	CPE4 elements, multistep, static analysis.
restart_psce4sh1_inter.inp	CPE4 elements, restart analysis.
xtcs4rs_interpolate.inp	S4R elements.

Stress analyses using Abaqus/Explicit:

xtc3d10ms_interpolate_xpl.inp	C3D10M elements.
xtcpe6ms_interpolate_xpl.inp	CPE6M elements.

Transferring temperatures between dissimilar meshes with user-specified regions

Problem description

The verification problems in this section test the interpolation of temperatures between dissimilar meshes with user-specified regions. The model consists of two part instances, as shown in *Figure 6*. A tiny gap exists between the two parts. A low gap heat transfer is applied along the gap so that a temperature jump results between the two adjacent surfaces. In this case temperature mapping using the general interpolation may result in erroneous temperature assignment to nodes on the adjacent surface due to the ambiguous association between target nodes near the interface surface and driving elements near this surface. The dissimilar mesh interpolation capability resolves the ambiguity by explicitly specifying the source regions in the heat transfer analysis from where the temperatures are read and the target regions in the current analysis onto which the temperatures are mapped.

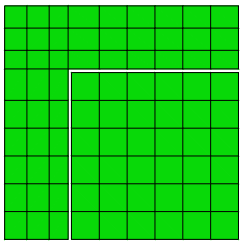


Figure 6: Model geometry with the gap amplified for illustration purposes.

Boundary conditions: The assembly is kept at a constant temperature of zero on the left boundary, and it is subjected to a constant surface heat flux of 0.003 on the right boundary and a constant surface heat flux of 1 on the top. The gap has a low gap heat conduction with a coefficient of 0.01.

Results and discussion

The nodes on the top boundary of the inner part in the current analysis are shifted up slightly so that they fall inside the outer part. This shift is done intentionally to illustrate a case that would result in incorrect driving element selection during interpolation. The mapped temperature results when the dissimilar mesh interpolation capability is not specified are shown in *Figure 7*. The test shows that the temperatures when regions are not specified for the interpolation of temperatures between dissimilar meshes are mapped incorrectly for those nodes on the top surface of the inner part. The error occurs because Abaqus searches for a parent element that encloses each node in the current analysis or is closest to each node. For the nodes on the top surface of the inner part, the parent elements are found inside the outer part, resulting in erroneous temperature definitions at the nodes. The mapped temperature results with the dissimilar mesh interpolation capability are shown in *Figure 8*. The capability fixes the error by explicitly specifying the source and the target regions of the interpolations. In this test case the driving element set from the previous heat transfer analysis is selected to cover the same instance region as that covered by the driven node set in the current analysis; therefore, instance-to-instance mapping is achieved.

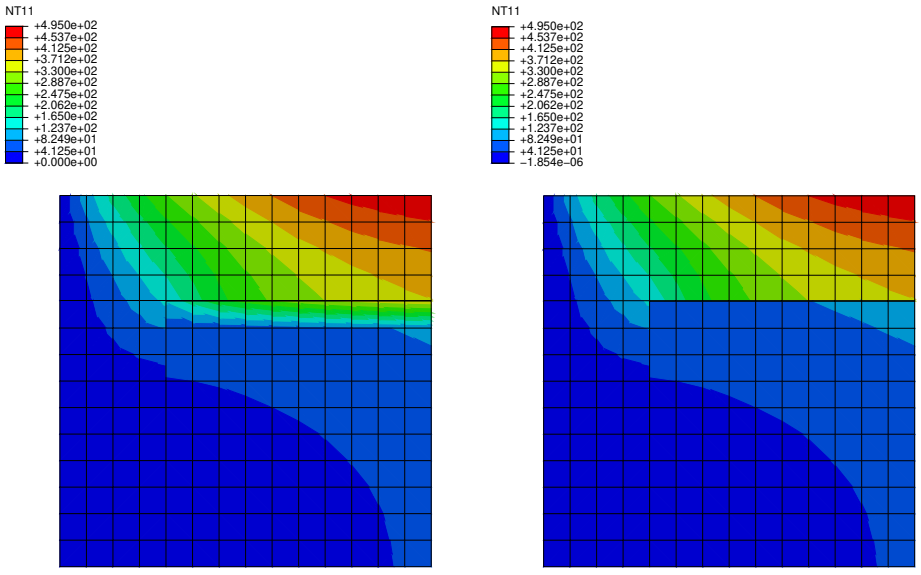


Figure 7: Temperature mapping without use of the dissimilar mesh interpolation capability.

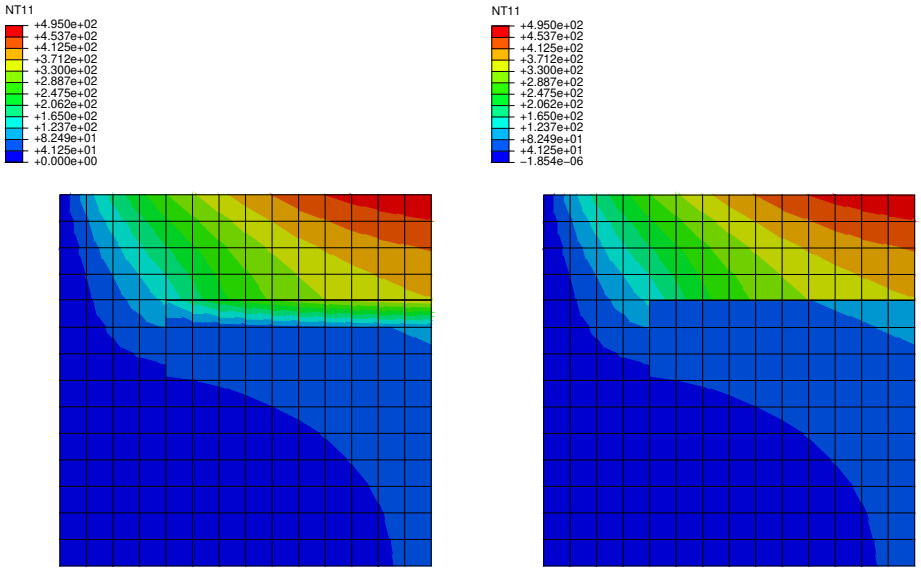


Figure 8: Temperature mapping with dissimilar mesh interpolation capability specified.

Input files

pgdc2d4.inp	Ancestor heat transfer analysis.
psdc2d4-no-drivingelsets.inp	Descendant model with no DRIVING ELSETS (Abaqus/Standard).
psdc2d4-drivingelsets.inp	Descendant model with DRIVING ELSETS (Abaqus/Standard).
psdc2d4-no-drivingelsets-xpl.inp	Descendant model with no DRIVING ELSETS (Abaqus/Explicit).
psdc2d4-drivingelsets-xpl.inp	Descendant model with DRIVING ELSETS (Abaqus/Explicit).

Reading temperature and pressure data from results files

Elements tested

CPE4 DC2D4

Problem description

These tests verify that temperatures and pressures are applied properly to a structure when various combinations of temperature and pressure stress values are used in a mass diffusion analysis. Temperature and pressure stress initial conditions are read from the results file of an Abaqus/Standard analysis, and a series of pressure and temperature loadings are applied to the nodes of an element using data line input in the following sequence:

- Step 1: Concentration ramped from 0 to 100 at a corner of the element.
- Step 2: A pressure gradient is applied along one diagonal of the element.
- Step 3: All pressures are reset to initial conditions.
- Step 4: A temperature gradient is applied along the same element diagonal as the pressure gradient in Step 2.
- Step 5: All temperatures are reset to initial conditions.
- Step 6: Pressure and temperature gradients are applied simultaneously along the element diagonal.

The material properties of the problem are defined such that

$$\kappa_s = -\kappa_p \left(\theta - \theta_z\right) \frac{\partial p}{\partial x}.$$

When both the temperature and pressure gradients are applied to the model, the diffusion is driven by concentration gradients alone.

The following must be confirmed by this test:

- Pressures must be set correctly using an amplitude curve.
- If existing values are removed, temperatures/pressures must be ramped back to their initial conditions or set to the new values defined on the data lines.

Results and discussion

The results match the exact analytical solutions for the applied temperature and pressure gradients.

Input files

xpressic.inp	This analysis generates a results file with temperature and pressure stress data, which is used to define initial conditions in xpresspt.inp.
xpresspt.inp	This file tests the setting of temperature and pressure using data line input, as outlined earlier.
xpressre.inp	This analysis restarts the second step of xpresspt.inp from increment 2. Results at the end of the analysis should be identical to the results at the end of the second step in xpresspt.inp.

Reading solution-dependent variables from results files

Elements tested

DC1D2 T3D2

Problem description

These tests verify that the solution-dependent variables from a heat transfer analysis are properly transferred as field variables in the subsequent stress analysis. The structure being analyzed is a cantilevered truss made up of 10 one-dimensional link elements. The solution-dependent state variables written to the results file are the averages of the values extrapolated to the element nodes. A separate results file is then generated, where the solution-dependent state variables value is stored as the second attribute under record key 201.

The temperature and field variable values are set by reading the data from the results file of the heat transfer run as follows:

xsdvttrt.fil ⇒ Temperature

xsdvttrt1.fil ⇒ Field variable

Results and discussion

The solution-dependent variable is transferred correctly into the stress analysis as a field variable.

Input files

xsdvttrt.inp	Heat transfer analysis.
xsdvttrt.f	User subroutine *HETVAL* used in xsdvttrt.inp.
xsdvttrs.inp	Stress analysis.
xsdvt.f	Postprocessing program.

Reading scalar nodal output from the output database into field variables

Elements tested

CPE3 CPE4 CPE4R CPE6 CPE6M CPE8 CPE8R
C3D4 C3D6 C3D8 C3D10 C3D10M C3D15 C3D20R

Problem description

These tests verify that Abaqus/Standard:

- can properly read and interpolate nodal output variables such as temperature, normalized concentrations, and electric potential fields from an output database. Further, these are initialized and defined as field variables in a subsequent analysis using the initial conditions, temperature, and field variable procedures;
- can read and interpolate results correctly from different analyses and meshes; and
- can allow a combination of volumetric expansion terms driven by temperature and/or field variables in the same material definition.

The basic test procedure is as follows: A set of initial two- and three-dimensional heat transfer, mass diffusion, and piezoelectric analyses are run. In these analyses temperatures, normalized concentrations, and electric potentials are written as nodal data to output databases. Different combinations of temperature, normalized concentrations, and electric potential fields are read from these analyses and used to initialize and define temperature and field variables in subsequent stress/displacement analyses. Using the thermal and field expansion capability in Abaqus/Standard, the temperatures and field variables are used to drive the displacement fields by imposing volumetric strains.

Results and discussion

The tests verify that temperature, normalized temperature, and electric potential fields are properly read and interpolated from an output database to initialize and define field variables.

Input files

heattransfer2d.inp	Two-dimensional heat transfer analysis using different continuum heat transfer elements; temp_nnc_epot.f is used to drive the temperatures.
heattransfer_dc2d4.inp	Two-dimensional heat transfer analysis using DC2D4 heat transfer elements; temp_nnc_epot.f is used to drive the temperatures.
heattransfer3d.inp	Three-dimensional heat transfer analysis using different continuum heat transfer elements; temp_nnc_epot.f is used to drive the temperatures.
heattransfer_dc3d8.inp	Three-dimensional heat transfer analysis using DC3D8 heat transfer elements; temp_nnc_epot.f is used to drive the temperatures.
massdiffusion2d.inp	Two-dimensional mass diffusion analysis using different continuum mass diffusion elements; temp_nnc_epot.f is used to drive the normalized concentrations.

massdiffusion_dc2d4.inp	Two-dimensional mass diffusion analysis using DC2D4 mass diffusion elements; temp_nnc_epot.f is used to drive the normalized concentrations.
massdiffusion3d.inp	Three-dimensional mass diffusion analysis using different continuum mass diffusion elements; temp_nnc_epot.f is used to drive the normalized concentrations.
massdiffusion_dc3d8.inp	Three-dimensional mass diffusion analysis using DC3D8 mass diffusion elements; temp_nnc_epot.f is used to drive the normalized concentrations.
piezoelectric2d.inp	Two-dimensional piezoelectric analysis using different continuum piezoelectric elements; temp_nnc_epot.f is used to drive the electric potentials.
piezoelectric_cpe4e.inp	Two-dimensional piezoelectric analysis using CPE4E piezoelectric elements; temp_nnc_epot.f is used to drive the electric potentials.
piezoelectric3d.inp	Three-dimensional piezoelectric analysis using different continuum piezoelectric elements; temp_nnc_epot.f is used to drive the electric potentials.
piezoelectric_c3d8e.inp	Three-dimensional piezoelectric analysis using C3D8E piezoelectric elements; temp_nnc_epot.f is used to drive the electric potentials.
temp_nnc_epot.f	User subroutine *DISP* used to drive heat transfer, mass diffusion, and piezoelectric analyses.
static_temp_2d.inp	Two-dimensional stress analysis that has volumetric expansion coming from temperatures. The temperature field is read from heattransfer2d.inp.
static_nnc_2d.inp	Two-dimensional stress analysis that has volumetric expansion coming from normalized concentrations. The normalized concentration field is read from massdiffusion2d.inp.
static_epot_2d.inp	Two-dimensional stress analysis that has volumetric expansion coming from electric potentials. The electric potential field is read from piezoelectric2d.inp.
static_temp_nnc_2d.inp	Two-dimensional stress analysis that has volumetric expansion coming from temperatures and normalized concentrations. The temperature field is read from heattransfer2d.inp, and the normalized concentration field is read from massdiffusion2d.inp.
static_temp_epot_2d.inp	Two-dimensional stress analysis that has volumetric expansion coming from temperatures and electric potentials. The temperature field is read from heattransfer2d.inp, and the electric potential field is read from piezoelectric2d.inp.
static_nnc_epot_2d.inp	Two-dimensional stress analysis that has volumetric expansion coming from normalized concentrations and electric potentials. The normalized concentration field is read from massdiffusion2d.inp, and the electric potential field is read from piezoelectric2d.inp.
static_temp_nnc_epot_2d.inp	Two-dimensional stress analysis that has volumetric expansion coming from temperatures, normalized concentrations, and electric potentials. The temperature field is read from heattransfer2d.inp, the normalized concentration field is read from massdiffusion2d.inp, and the electric potential field is read from piezoelectric2d.inp.
static_temp_interp_2d.inp	Two-dimensional stress analysis that has volumetric expansion coming from temperatures. The temperature field is read from heattransfer_dc2d4.inp. Temperatures are interpolated from two-dimensional quadrilateral meshes.
static_nnc_interp_2d.inp	Two-dimensional stress analysis that has volumetric expansion coming from normalized concentrations. The normalized concentration field is read from massdiffusion_dc2d4.inp. Normalized concentrations are interpolated from two-dimensional quadrilateral meshes.
static_epot_interp_2d.inp	Two-dimensional stress analysis that has volumetric expansion coming from electric potentials. The electric potential field is read from piezoelectric_cpe4e.inp. Electric potentials are interpolated from two-dimensional quadrilateral meshes.

static_temp_nnc_interp_2d.inp	Two-dimensional stress analysis that has volumetric expansion coming from temperatures and normalized concentrations. The temperature field is read from heattransfer_dc2d4.inp, and the normalized concentration field is read from massdiffusion_dc2d4.inp. All fields are interpolated from two-dimensional quadrilateral meshes.
static_temp_epot_interp_2d.inp	Two-dimensional stress analysis that has volumetric expansion coming from temperatures and electric potentials. The temperature field is read from heattransfer_dc2d4.inp, and the electric potential field is read from piezoelectric_cpe4e.inp. All fields are interpolated from two-dimensional quadrilateral meshes.
static_nnc_epot_interp_2d.inp	Two-dimensional stress analysis that has volumetric expansion coming from normalized concentrations and electric potentials. The normalized concentration field is read from massdiffusion_dc2d4.inp, and the electric potential field is read from piezoelectric_cpe4e.inp. All fields are interpolated from two-dimensional quadrilateral meshes.
static_temp_nnc_epot_interp_2d.inp	Two-dimensional stress analysis that has volumetric expansion coming from temperatures, normalized concentrations, and electric potentials. The temperature field is read from heattransfer_dc2d4.inp, the normalized concentration field is read from massdiffusion_dc2d4.inp, and the electric potential field is read from piezoelectric_cpe4e.inp. All fields are interpolated from two-dimensional quadrilateral meshes.
static_temp_3d.inp	Three-dimensional stress analysis that has volumetric expansion coming from temperatures. The temperature field is read from heattransfer3d.inp. Temperatures are interpolated from three-dimensional hexahedral meshes.
static_nnc_3d.inp	Three-dimensional stress analysis that has volumetric expansion coming from normalized concentrations. The normalized concentration field is read from massdiffusion3d.inp.
static_nnc_3d_rs.inp	Restart analysis of static_nnc_3d.inp.
static_epot_3d.inp	Three-dimensional stress analysis that has volumetric expansion coming from electric potentials. The electric potential field is read from piezoelectric3d.inp.
static_temp_nnc_3d.inp	Three-dimensional stress analysis that has volumetric expansion coming from temperatures and normalized concentrations. The temperature field is read from heattransfer3d.inp, and the normalized concentration field is read from massdiffusion3d.inp.
static_temp_epot_3d.inp	Three-dimensional stress analysis that has volumetric expansion coming from temperatures and electric potentials. The temperature field is read from heattransfer3d.inp, and the electric potential field is read from piezoelectric3d.inp.
static_nnc_epot_3d.inp	Three-dimensional stress analysis that has volumetric expansion coming from normalized concentrations and electric potentials. The normalized concentration field is read from massdiffusion3d.inp, and the electric potential field is read from piezoelectric3d.inp.
static_temp_nnc_epot_3d.inp	Three-dimensional stress analysis that has volumetric expansion coming from temperatures, normalized concentrations, and electric potentials. The temperature field is read from heattransfer3d.inp, the normalized concentration field is read from massdiffusion3d.inp, and the electric potential field is read from piezoelectric3d.inp.
static_temp_nnc_epot_3d_rs.inp	Restart analysis of static_temp_nnc_epot_3d.inp.
static_temp_interp_3d.inp	Three-dimensional stress analysis that has volumetric expansion coming from temperatures. The temperature field is read from heattransfer_dc3d8.inp. Temperatures are interpolated from three-dimensional hexahedral meshes.

static_nnc_interp_3d.inp	Three-dimensional stress analysis that has volumetric expansion coming from normalized concentrations. The normalized concentration field is read from massdiffusion_dc3d8.inp. The normalized concentrations are interpolated from three-dimensional hexahedral meshes.
static_epot_interp_3d.inp	Three-dimensional stress analysis that has volumetric expansion coming from electric potentials. The electric potential field is read from piezoelectric_c3d8e.inp. The electric potentials are interpolated from three-dimensional hexahedral meshes.
static_epot_interp_3d_rs.inp	Restart analysis of static_epot_interp_3d.inp.
static_temp_epot_interp_3d.inp	Three-dimensional stress analysis that has volumetric expansion coming from temperatures and electric potentials. The temperature field is read from heattransfer_dc3d8.inp, and the electric potential field is read from piezoelectric_c3d8e.inp. All fields are interpolated from three-dimensional hexahedral meshes.
static_nnc_epot_interp_3d.inp	Three-dimensional stress analysis that has volumetric expansion coming from normalized concentrations and electric potentials. The normalized concentration field is read from massdiffusion_dc3d8.inp, and the electric potential field is read from piezoelectric_c3d8e.inp. All fields are interpolated from three-dimensional hexahedral meshes.
static_temp_nnc_epot_interp_3d.inp	Three-dimensional stress analysis that has volumetric expansion coming from temperatures, normalized concentrations, and electric potentials. The temperature field is read from heattransfer_dc3d8.inp, the normalized concentration field is read from massdiffusion_dc3d8.inp, and the electric potential field is read from piezoelectric_c3d8e.inp. All fields are interpolated from three-dimensional hexahedral meshes.
static_temp_nnc_epot_interp_3d_rs.inp	Restart analysis of static_temp_nnc_epot_interp_3d.inp.

Specifying a known predefined pore fluid pressure field

Elements tested

C3D4 C3D5 C3D8 CAX4R

Problem description

These tests verify that a pore fluid pressure field can be predefined, initialized, and properly applied to a stress analysis without the presence of pore fluid pressure degrees of freedom. The first test includes a constant predefined pore fluid field, and the other tests contain predefined pore fluid pressure fields that vary with position and time.

Results and discussion

In the constant pore fluid pressure test, the results agree with analytical solutions. In the varying pore fluid pressure tests, the mechanical responses agree with the results from the corresponding coupled pore fluid diffusion/stress analyses with the same pore fluid pressure fields.

Input files

predefined_por_std_c3d8_c3d4_c3d5.inp	Three-dimensional static stress analysis using different continuum solid elements. The predefined pore fluid pressure field is initialized. Its magnitude ramps down to zero in the first step and ramps up to 100 in the second step.

predefined_por_cax4r_elas_std.inp	Static stress analysis using axisymmetric elements.
predefined_por_cax4r_elas_xpl.inp	Explicit dynamic analysis using axisymmetric elements.
amp_por_elas.inp	Amplitude definitions used in both `predefined_por_cax4r_elas_std.inp` and `predefined_por_cax4r_elas_xpl.inp`.
predefined_por_cax4r_elas_xpl.f	User subroutines used in `predefined_por_cax4r_elas_xpl.inp`.
predefined_por_cax4r_crushablefoam_std.inp	Static stress analysis using axisymmetric elements.
predefined_por_cax4r_crushablefoam_xpl.inp	Explicit dynamic analysis using axisymmetric elements.
predefined_por_cax4r_crushablefoam_xpl_no_vusubs.inp	Explicit dynamic analysis using axisymmetric elements.
amp_por_crushablefoam.inp	Amplitude definitions used in `predefined_por_cax4r_crushablefoam_std.inp`, `predefined_por_cax4r_crushablefoam_xpl.inp`, and `predefined_por_cax4r_crushablefoam_xpl_no_vusubs.inp`.
predefined_por_cax4r_crushablefoam_xpl.f	User subroutines used in `predefined_por_cax4r_crushablefoam_xpl.inp`.

Coupled pore-thermal elements

Products: Abaqus/Standard

Overview

This problem contains basic test cases for one or more Abaqus elements and features.

Elements tested

C3D8PT C3D10MPT COD2D4PT COD3D8PT

Features tested

Tests are created to verify specialized aspects of coupled pore-thermal elements.

Problem description

Simple tests are created to test steady-state heat transfer, heat convection through pore fluid flow, use of latent heat, and solution mapping.

Results and discussion

In all cases the results are as expected.

Input files

c3d8pt_ss_dsflux.inp	Test for steady-state heat transfer.
ctup_latent_heat.inp	Test for use of latent heat in material properties to model phase change.
one_d_soil_convection.inp	Test for heat convection via pore fluid flow.
one_d_soil_convection.f	User subroutine used in one_d_soil_convection.inp.
pmap_c3d8pt_elastic_a.inp	First test to verify the solution mapping capability.
pmap_c3d10mpt_elastic_d.inp	Second test to verify the solution mapping capability.
advection_cod2d4pt.inp	Test for gap fluid advection using two-dimensional coupled temperature–pore pressure cohesive elements.
advection_cod3d8pt.inp	Test for gap fluid advection using three-dimensional coupled temperature–pore pressure cohesive elements.
film_diff_advection_cod3d8pt.inp	Test for gap fluid convection using three-dimensional coupled temperature–pore pressure cohesive elements.

Product Index

This index lists the sections in this guide that apply to a particular Abaqus product.

In this section:

- *Abaqus/Standard*
- *Abaqus/Explicit*
- *Abaqus/CAE*
- *Abaqus/AMS*
- *Abaqus/Aqua*
- *Abaqus/Design*

Abaqus/Standard

This section lists the topics in this guide that apply to Abaqus/Standard.

Abaqus/Aqua analysis

Aqua load cases

Elastic-plastic joint elements

Jack-up foundation analysis

Acoustic and shock analyses

Adaptive meshing applied to coupled structural-acoustic problems

Impedance boundary conditions

Perfectly matched layers

Transient acoustic wave propagation

Volumetric drag

Activation functions used by feed-forward neural networks

Activation functions used by feed-forward neural networks

Adaptive mesh for solid elements in Abaqus/Standard

Adaptive mesh for solid elements in Abaqus/Standard

Adaptive remeshing

Error indicators

Pressurized thick-walled cylinder

Additive Manufacturing Process Simulation

Heat energy balance

Material deposition

Material removal

Metallurgical phase transformations

Moving heat sources

Pattern-based eigenstrain analysis

Trajectory-based eigenstrain analysis

Adjusting nodal coordinates

Adjusting nodal coordinates

Amplitude

Amplitude

Connector element verification

Connector elements in perturbation analyses

Damped free vibration with initial conditions

Individual connector option tests

Multiple instances of connector elements

Sinusoidal excitation of a damped spring-mass system

Tests for special-purpose connectors

Co-simulation

Abaqus/Standard to Abaqus/Explicit co-simulation

Fluid-structure interaction of a cantilever beam inside a channel

Coupled pore-thermal elements

Coupled pore-thermal elements

Coupled thermal-electrical analysis

Fully coupled thermal-electrochemical procedures

Modeling a half-cell lithium metal anode battery with multiphysics Butler-Volmer surface loading

Modeling a solid electrolyte and solid-state battery

Crack propagation

Crack propagation analysis

Hydraulically driven branching fracture

Modeling slurry transport and placement in hydraulically driven fracture

Propagation of hydraulically driven fracture

Define a part instance that will be used for display only

Define a part instance that will be used for display only

Defining a matrix for part of a model

Defining a matrix for part of a model

Defining temperature, field variable, and pressure stress values

Defining temperature, field variable, pressure stress, and pore fluid pressure values

Defining the cutting surface

Defining the cutting surface

Dynamic analysis

Modal dynamic analysis with baseline correction

Random response analysis

Steady-state dynamic analysis for infinite elements

Steady-state dynamic analysis for two-dimensional elements

Eigenvalue tests

Acoustic modes

Eigenvalue extraction for single unconstrained elements

Eigenvalue extraction for unconstrained patches of elements

Electromagnetic analysis

Eddy current analysis

Magnetostatic analysis

Element loading options

Beam stress/displacement elements

Cohesive element load verification

Continuum and shell heat transfer elements

Continuum mass diffusion elements

Continuum pore pressure elements

Continuum stress/displacement elements

Coupled temperature-displacement elements

Coupled thermal-electrical-structural elements

Distributed traction and edge loads

ELBOW elements

Incident wave loading

Mass and rotary inertia elements

Piezoelectric elements

Pipe stress/displacement elements

Rigid elements

Shell, membrane, and truss stress/displacement elements

Surface tension loads

Thermal-electrical elements

Element nodal forces in beam section orientation

Element nodal forces in beam section orientation

Establishing geostatic equilibrium when the initial stress state is unknown

Establishing geostatic equilibrium when the initial stress state is unknown

Extrapolation

Extrapolation

Inertia-based load balancing

Inertia-based load balancing

Mechanical properties

Annealing temperature

Barlat anisotropic plasticity

Composite, mass proportional, and rotary inertia proportional damping in Abaqus/Standard

Concrete damaged plasticity

Concrete smeared cracking

Cure modeling in Abaqus/Standard

Elastic materials

Electrical and mechanical modeling of cardiac tissue

Field expansion test

Generalized damage and failure for anisotropic materials

Hydrostatic fluid

Hysteretic materials

Johnson-Cook plasticity

Mass proportional damping

Material damping

Mullins effect and permanent set

Nonlinear large-strain viscoelasticity with hyperelasticity

Paperboard material model

Plasticity corrections

Progressive damage and failure in fiber-reinforced materials

Progressive damage and failure of ductile metals

Rate-dependent plasticity in Abaqus/Standard

Rate-independent plasticity

Structural damping

Thermal expansion test in Abaqus/Standard

Transient internal pressure loading of a viscoelastic cylinder

Two-layer viscoplasticity

Viscoelastic materials

Meshed beam cross-sections

Meshing and analyzing a two-dimensional model of a beam cross-section

Using generated cross-section properties in a beam analysis

Miscellaneous procedures and techniques

Complex eigenvalue extraction

Controlling material rate dependence

Direct cyclic and low-cycle fatigue analyses

Frequency extraction using the AMS eigensolver

Media transport

One-step inverse analysis

Steady-state dynamics with nondiagonal damping using the AMS eigensolver

Transferring results between dissimilar meshes in Abaqus/Standard

Miscellaneous tests

Analysis of unbounded acoustic regions

Bending stabilization for full-integration linear elements

Continuum shells: basic element modes

Convection elements: transport of a temperature pulse

Linear dynamic analysis with fluid link

Nonstructural mass verification

One-dimensional steady-state dynamic solutions for poroelastic acoustic elements

Rebar in Abaqus/Standard

Rigid bodies with temperature DOFs, heat capacitance, and nodal-based thermal loads

Transverse shear for shear-flexible shells

Model change

Acoustic model change: steady state

Contact model change

Coupled temperature-displacement model change: steady state

Heat transfer model change: steady state

Pore-thermal model change

Stress/displacement model change: dynamic

Stress/displacement model change: general tests

Stress/displacement model change: static

Patch tests

Membrane patch test

Patch test for acoustic elements

Patch test for axisymmetric elements

Patch test for axisymmetric elements with twist

Patch test for beam elements

Patch test for cylindrical elements

Patch test for heat transfer elements

Patch test for plate bending

Patch test for thermal-electrical elements

Patch test for three-dimensional solid elements

Piezoelectric analysis

Frequency extraction analysis for piezoelectric materials

General analysis procedures for piezoelectric materials

Static analysis for piezoelectric materials

Piezoresistivity properties: Piezoresistive effect

Piezoresistivity properties: Piezoresistive effect

Pre-tension section

Pre-tension section

Radiation view factor definition: symmetries and blocking

Radiation view factor definition: symmetries and blocking

Release rotational degrees of freedom

Release rotational degrees of freedom

Rigid body motion output variables

Rigid body motion output variables

Simple load tests

Axisymmetric membrane elements

Axisymmetric solid elements

Axisymmetric solid elements with twist

Beam added inertia

Beam fluid inertia

Cantilever sandwich beam: shear flexible shells

Cohesive elements

Constant curvature test for shells

Coriolis loading for direct-solution steady-state dynamic analysis

Cylinder subjected to an asymmetric pore pressure field: CAXA elements

Cylinder subjected to an asymmetric temperature field: CAXA elements

Cylinder subjected to asymmetric pressure loads: CAXA elements

Cylindrical elements

Cylindrical membrane elements

Flexure of a deep beam

Fluid link element

Fluid pipe element

Gasket behavior verification

Gasket element assembly

Generalized plane strain elements with relative motion of bounding planes

Hydrostatic fluid elements

Initial curvature of beams and shells

Loading of piezoelectric elements

Love-Kirchhoff beams and shells

Membrane loading of plane stress, plane strain, membrane, and shell elements

Modal dynamic and transient dynamic analysis with CAXA and SAXA elements

Normal definitions of beams and shells

Pipe-soil interaction elements

Pure bending of a cylinder: CAXA elements

Shear flexible beams and shells: I

Shear flexible beams and shells: II

Shear panel elements

Shell offset

Simple load tests for thermal-electrical elements

Surface-based pressure penetration

Temperature-dependent film condition

Thermal fluid pipe elements

Thermal stress in a cylindrical shell

Three-bar truss

Three-dimensional solid elements

Variable thickness shells and membranes

Verification of beam elements and section types

Verification of section forces for shells

Verification of the elastic behavior of frame elements

Verification of the plastic behavior of frame elements

Spatially varying element properties

Spatially varying element properties

Special-purpose stress/displacement elements

Distributing coupling elements

Drag chain elements

Flexible joint element

Line spring elements

Specifying geometric imperfection and parameter shape variation

Specifying geometric imperfection and parameter shape variation

Specifying user-defined field variables directly in the input file

Specifying user-defined field variables directly in the input file

Submodeling

Acoustic submodeling

Axisymmetric continuum stress/displacement submodeling

Axisymmetric stress/displacement submodeling with twist

Coupled temperature-displacement submodeling

Cylindrical continuum stress/displacement submodeling

Gasket submodeling

General submodeling

Heat transfer submodeling

Membrane submodeling

Miscellaneous submodeling tests

Piezoelectric submodeling

Pore pressure submodeling

Shell submodeling

Shell-to-solid submodeling

Surface element submodeling

Three-dimensional continuum stress/displacement submodeling

Two-dimensional continuum stress/displacement submodeling

Substructure matrix output request

Substructure matrix output request

Substructuring

Coupled structural-acoustic analysis with substructures

Degenerated elements within a substructure

Frequency extraction for substructures

Substructure damping

Substructure load case with centrifugal loads

Substructure load case with nodal pressure loads

Substructure preload history

Substructure recovery with The nodal Transformation

Substructure removal

Substructure rotation, mirroring, transformation, and constraints

Substructures with large rotations

Substructures with rebar

Thermal-stress analysis with substructures

Symmetric model generation and analysis of cyclic symmetry models

Analysis of cyclic symmetric models

Symmetric model generation and results transfer

Thermal properties

Conductivity and specific heat

Field-variable-dependent conductivity

Gap conductance

Transferring results between Abaqus/Standard and Abaqus/Explicit

Adding and removing elements during results transfer

Changing the material definition during import

Miscellaneous results transfer tests

Transferring connector elements into Abaqus/Explicit

Transferring hourglass forces

Transferring mass and rotary inertia elements

Transferring results between Abaqus/Explicit and Abaqus/Standard

Transferring results for a hyperelastic sheet with a circular hole

Transferring results from one Abaqus/Standard analysis to another Abaqus/Standard analysis

Transferring results with damage

Transferring results with general beam sections

Transferring results with general shell sections

Transferring results with hyperelasticity

Transferring results with hyperfoam

Transferring results with orientation

Transferring results with parallel rheological framework

Transferring results with plasticity

Transferring results with viscoelasticity

Transferring rigid elements

User subroutine verification

DFLUX

DISP

DLOAD

FRIC

FRIC_COEF

GAPCON

GAPELECTR

HARDINI

HETVAL

RSURFU

SDVINI

UAMP

UANISOHYPER_INV and VUANISOHYPER_INV

UCOHESIVEOFFSET

UCREEPNETWORK

UCURE

UEL

UELMAT

UEPACTIVATIONVOL

UEXPAN

UFLUID

UGENS

UHARD

UHYPER_STRETCH

UINTER

UMAT and UHYPER

UMATHT

URDFIL

USDFLD

UTEMP, VUTEMP, UFIELD, UMASFL, and UPRESS

UTRSNETWORK

UVAREL

UVARM

UWAVE and UEXTERNALDB

User-defined coordinate system

User-defined coordinate system

Writing element matrix output records

Writing element matrix output records

Abaqus/Explicit

This section lists the topics in this guide that apply to Abaqus/Explicit.

Abaqus/Aqua analysis

Aqua load cases

Acoustic and shock analyses

Blast loading of a circular plate using the CONWEP model

CONWEP blast loading pressures

Impedance boundary conditions

Transient acoustic wave propagation

Volumetric drag

Adjusting nodal coordinates

Adjusting nodal coordinates

Amplitude

Amplitude

Connector element verification

Damped free vibration with initial conditions

Individual connector option tests

Multiple instances of connector elements

Sinusoidal excitation of a damped spring-mass system

Tests for special-purpose connectors

Co-simulation

Abaqus/Standard to Abaqus/Explicit co-simulation

Define a part instance that will be used for display only

Define a part instance that will be used for display only

Defining temperature, field variable, and pressure stress values

Defining temperature, field variable, pressure stress, and pore fluid pressure values

Defining the cutting surface

Defining the cutting surface

Discrete particle methods

Discrete element method analysis

Discrete particle swelling

Lumped kinetic molecular method

Particle generator

Dynamic analysis

Linear kinematics element tests

Mass scaling

Single degree of freedom spring-mass systems

Element loading options

Abaqus/Explicit element loading verification

Cohesive element load verification

Coupled temperature-displacement elements

Distributed traction and edge loads

Incident wave loading

Pipe stress/displacement elements

Surface tension loads

Integrated output variables

Integrated output variables

Mechanical properties

Annealing temperature

Barlat anisotropic plasticity

Brittle cracking constitutive model

Concrete damaged plasticity

Cracking model: tension shear test

Drucker-Prager plasticity

Drucker-Prager/Cap plasticity model

Elastic materials

Electrical and mechanical modeling of cardiac tissue

Equation of state material

Field-variable-dependent elastic materials

Field-variable-dependent inelastic materials

Generalized damage and failure for anisotropic materials

Johnson-Cook plasticity

Large-strain viscoelasticity with hyperelasticity

Mass proportional damping

Material damping

Mullins effect and permanent set

Nonlinear large-strain viscoelasticity with hyperelasticity

Paperboard material model

Porous metal plasticity

Progressive damage and failure in fiber-reinforced materials

Progressive damage and failure of ductile metals

Rate-dependent plasticity in Abaqus/Explicit

Rate-independent plasticity

Temperature-dependent elastic materials

Temperature-dependent inelastic materials

Thermal expansion test in Abaqus/Explicit

Viscoelastic materials

Meshed beam cross-sections

Using generated cross-section properties in a beam analysis

Miscellaneous procedures and techniques

CEL analysis of a rotating water disk

Controlling material rate dependence

Media transport

Ramping of initial stress

Smoothed particle hydrodynamic analysis

Undrained pore fluid flow and stress analysis

Miscellaneous tests

Analysis of unbounded acoustic regions

Beam rebar line reinforcement in Abaqus/Standard and Abaqus/Explicit

Defining a reference mesh

Mass adjust verification

Nonstructural mass verification

Rebar in Abaqus/Explicit

Rigid bodies with temperature DOFs, heat capacitance, and nodal-based thermal loads

Patch tests

Membrane patch test

Patch test for acoustic elements

Patch test for axisymmetric elements

Patch test for beam elements

Patch test for heat transfer elements

Patch test for three-dimensional solid elements

Simple load tests

Beam added inertia

Beam fluid inertia

Beam with end moment

Cohesive elements

Composite shell sections

Flexure of a deep beam

Initial curvature of beams and shells

Normal definitions of beams and shells

Shear flexible beams and shells: I

Shear flexible beams and shells: II

Simple shear

Simple tests of beam kinematics

Surface-based pressure penetration

Temperature-dependent film condition

Tensile test

Thermal stress in a cylindrical shell

Spatially varying element properties

Spatially varying element properties

Specifying geometric imperfection and parameter shape variation

Specifying geometric imperfection and parameter shape variation

Specifying user-defined field variables directly in the input file

Specifying user-defined field variables directly in the input file

Submodeling

Acoustic submodeling

Axisymmetric continuum stress/displacement submodeling

Coupled temperature-displacement submodeling

General submodeling

Heat transfer submodeling

Membrane submodeling

Miscellaneous submodeling tests

Shell submodeling

Shell-to-solid submodeling

Three-dimensional continuum stress/displacement submodeling

Two-dimensional continuum stress/displacement submodeling

Surface-based fluid cavities

Surface-based fluid cavities

Symmetric model generation and analysis of cyclic symmetry models

Analysis of cyclic symmetric models

Thermal properties

Conductivity and specific heat

Field-variable-dependent conductivity

Gap conductance

Transferring results between Abaqus/Standard and Abaqus/Explicit

Adding and removing elements during results transfer

Changing the material definition during import

Miscellaneous results transfer tests

Transferring connector elements into Abaqus/Explicit

Transferring hourglass forces

Transferring mass and rotary inertia elements

Transferring results between Abaqus/Explicit and Abaqus/Standard

Transferring results for a hyperelastic sheet with a circular hole

Transferring results from multiple previous analyses

Transferring results from one Abaqus/Explicit analysis to another Abaqus/Explicit analysis

Transferring results of an analysis model multiple times

Transferring results with damage

Transferring results with general beam sections

Transferring results with general shell sections

Transferring results with hyperelasticity

Transferring results with hyperfoam

Transferring results with nodal temperature and field variables

Transferring results with orientation

Transferring results with parallel rheological framework

Transferring results with plasticity

Transferring results with viscoelasticity

Transferring rigid elements

User subroutine verification

UANISOHYPER_INV and VUANISOHYPER_INV

UTEMP, VUTEMP, UFIELD, UMASFL, and UPRESS

VDISP

VDLOAD

VFILM

VFRIC, VFRIC_COEF, and VFRICTION

VHETVAL

VSDVINI

VUAMP

VUCHARLENGTH

VUCREEPNETWORK

VUEL

VUFIELD

VUGENS

VUHARD

VUHYPER

VUHYPER_STRETCH

VUINTER

VUINTERACTION

VUMAT

VUMATHT

VUSDFLD

VUTRSNETWORK

VUVISCOSITY

VWAVE

Abaqus/CAE

This section lists the topics in this guide that apply to Abaqus/CAE.

Adaptive remeshing

Pressurized thick-walled cylinder

Abaqus/AMS

This section lists the topics in this guide that apply to Abaqus/AMS.

Miscellaneous procedures and techniques

Steady-state dynamics with nondiagonal damping using the AMS eigensolver

Abaqus/Aqua

This section lists the topics in this guide that apply to Abaqus/Aqua.

Abaqus/Aqua analysis

Aqua load cases

Elastic-plastic joint elements

Jack-up foundation analysis

User subroutine verification

VWAVE

Abaqus/Design

This section lists the topics in this guide that apply to Abaqus/Design.

Design sensitivity analysis

Design sensitivity analysis controls

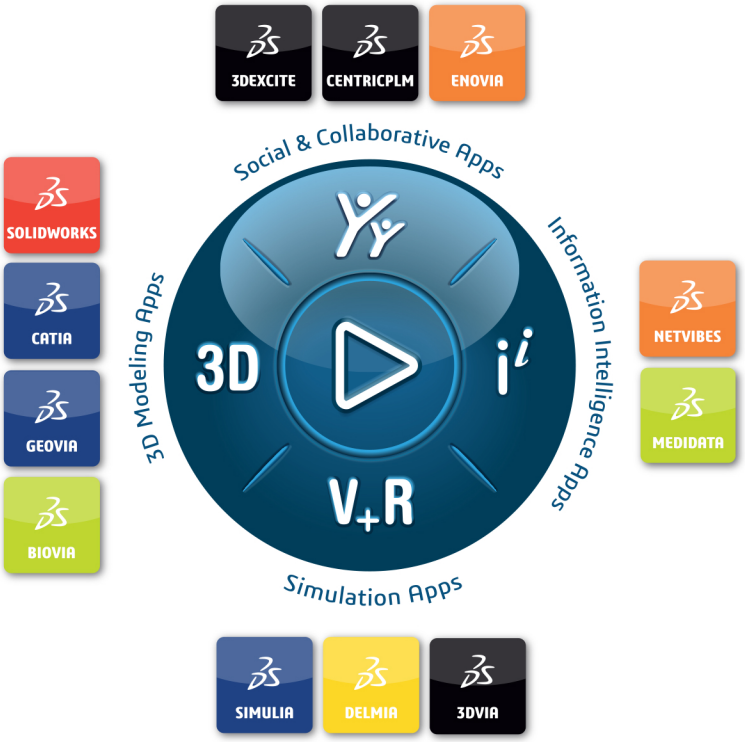
Design sensitivity analysis for contact

Design sensitivity analysis for continuum elements

Design sensitivity analysis for elastic materials

Design sensitivity analysis for special-purpose elements

Design sensitivity analysis for structural elements



Our **3D**EXPERIENCE® platform powers our brand applications, serving 12 industries, and provides a rich portfolio of industry solution experiences.

Dassault Systèmes is a catalyst for human progress. We provide business and people with collaborative virtual environments to imagine sustainable innovations. By creating virtual twin experiences of the real world with our **3D**EXPERIENCE platform and applications, our customers can redefine the creation, production and life-cycle-management processes of their offer and thus have a meaningful impact to make the world more sustainable. The beauty of the Experience Economy is that it is a human-centered economy for the benefit of all – consumers, patients and citizens.

Dassault Systèmes brings value to more than 300,000 customers of all sizes, in all industries, in more than 150 countries. For more information, visit **www.3ds.com**.

Europe/Middle East/Africa
Dassault Systèmes
10, rue Marcel Dassault
CS 40501
78946 Vélizy-Villacoublay Cedex
France

Asia-Pacific
Dassault Systèmes
17F, Foxconn Building,
No. 1366, Lujiazui Ring Road
Pilot Free Trade Zone, Shanghai 200120
China

Americas
Dassault Systèmes
175 Wyman Street
Waltham, Massachusetts
02451-1223
USA

DASSAULT SYSTEMES

©2026 Dassault Systèmes. All rights reserved. **3DEXPERIENCE**, the 3DS logo, the Compass icon, IFWE, 3DEXCITE, 3DVIA, BIOVIA, CATIA, CENTRIC PLM, DELMIA, ENOVIA, GEOVIA, MEDIDATA, NETVIBES, OUTSCALE, SIMULIA and SOLIDWORKS are commercial trademarks or registered trademarks of Dassault Systèmes, a European company (Societas Europaea) incorporated under French law, and registered with the Versailles trade and companies registry under number 322 306 440, or its subsidiaries in the United States and/or other countries.