True*Grid*®Output Manual For ABAQUS®

A Guide and a Reference

by

Robert Rainsberger

Version 2.3.0

XYZ Scientific Applications, Inc.

May 8, 2009

Table of Contents

I. ABAQUS® Output Guide

This manual teaches the use of **True*Grid*®** when applied to a model to serve as input to the ABAQUS® finite element simulation code. More specifically, this manual discusses the use of commands in **True*Grid*®** to produce material models, element types, boundary conditions, loads, procedures, steps, postprocessing options, and contact (sliding) surfaces that are uniquely designed for the ABAQUS output option in **True*Grid*®**. The meaning and purpose of these features within ABAQUS are not discussed here and this manual is not a substitute for the ABAQUS User's Manual. You should have some familiarity with the use of ABAQUS when using the features discussed in this manual. Also, the generic generation of the geometric model is covered extensively in the **True*Grid*®** User's Manual and is not repeated here.

Font Conventions

Different fonts are used through out this manual to indicate their meaning. A literal is highlighted in bold. A symbol to be substituted with a literal or a number is *italicized*. A computer example uses the `Courier font`. A button in from the Graphical User Interface is both ***italic and bold***.

Supported Features

There are many features in **True*Grid*®** to create a complete model for ABAQUS. The table below shows the commands that are used for each feature. Sometimes there are several commands listed. For example, shells can be generated using both the **block** and **cylinder** commands. The **n** and **th** are used to set the properties of these shells. In another example, the **si** and **sii** commands are used to identify the faces of the mesh that form the sliding (or contact) surfaces. The associated **sid** command is used to assign properties to the sliding surface.

ABAQUS feature	**True*Grid*®** commands
title to the problem	**title**
2nd order elements	**quadratic**
1st order elements	**linear**
beams	**bsd, ibm, ibmi, jbm, jbmi, kbm, kbmi, bm**
assign material properties	**mt**, **mti**, **mate**, **mtv**, **abaqmats**
nodal constraints	**b**, **bi**, **plane**
symmetry	**plane**
rigid body rotation	**rotation**
rigid body velocity	**velocity**
initial velocities	**ve, vei**
amplitude curves	**lcd**, **flcd**

shared nodal degrees of freedom	**jd**, **jt**
contact (sliding) interface	**sid**, **si**, **sii**
default initial nodal temperature	**temp**
initial nodal temperatures	**tm**, **tmi**
pressure faces	**pr**, **pri**
pressure amplitude	**pramp**, **dom**
concentrated nodal forces	**fc**, **fci, fcc, fcci, fcs, fcsi**
nodal displacements	**fd**, **fdi, fdc, fdci, fds, fdsi**
nodal velocities	**fv, fvi, fvc, fvci, fvs, fvsi, fvv, fvvi, fvvc, fvvci, fvvs, fvvsi**
nodal temperatures	**ft, fti,vft, vfti**
nodal accelerations	**acc, acci, accc, accci, accs, accsi, vacc, vacci, vaccc, vaccci, vaccs, vaccsi**
nodal moments	**mom, momi**
nodal mass	**pm**, **npm**
nodal sets	**nset**, **nseti**
element sets	**eset, eseti**
shared nodal (multiple point) constraints	**mpc**
offset the numbering of nodes and elements	**offset**
ABAQUS® output format	**abaqus**
time/history steps	**abaqstep**
element and material properties	**abaqmats**

You may want to view some of the properties graphically using the **condition** (**co**) command in the merge phase. The **tmm** command can be used to calculate the mass of each part. Be sure to merge the nodes using one of the merging commands such as **stp** and, finally, use the **abaqus** command to select ABAQUS® as the output option and the **write** command to actually create the input deck for ABAQUS®.

These commands will generate the following "*" keywords:

*ACOUSTIC MEDIUM	**aqacm** (**abaqmats** option)
*AMPLITUDE	**lcd, flcd**
*BEAM SECTION	**bsd** (see also **ibm, ibmi, jbm, jbmi, kbm, kbmi, bm**)
*BOUNDARY	**b, bi, plane, fd, fdi, fdc, fdci, fds, fdsi, fv, fvi, fvc, fvci, fvs, fvsi, fvv, fvvi, fvvc, fvvci, fvvs, fvvsi, ft, fti,vft, vfti, acc, acci, accc, accci, accs, accsi, vacc, vacci, vaccc, vaccci, vaccs, vaccsi**
*BUCKLE	**buckle** (**aqstep** option)
*CLAY PLASTICITY	**aqclay** (**abaqmats** option)
*CLOAD	**fc**, **fci, fcc, fcci, fcs, fcsi**, **mom, momi**
*CONCRETE	**aqconc** (**abaqmats** option)

*CONDUCTIVITY	**aqcond** (**abamats** option)
*CONTACT FILE	**crsltf** (**abstep** option)
*CONTACT NODE SET	**si, sii** (see also **sid**)
*CONTACT OUTPUT	**fielo, histo** (**abstep** options)
*CONTACT PAIR	**sid** (see also **si, sii**)
*CONTACT PRINT	**cdataf** (**abstep** option)
*COUPLED TEMPERATURE-DISPLACEMENT	**ctd** (**abstep** option)
*CREEP	**aquscre** (**abaqmats** option)
*CYCLED PLASTIC	**aqcycl** (**abaqmats** option)
*DAMPING	**aqfcdf, aqspdf, aqmpmf** (**abaqmats** option)
*DEFORMATION PLASTICITY	**aqdepl** (**abaqmats** option)
*DENSITY	**aqdens** (**abaqmats** option)
*DEPVAR	**aqdepv** (**abaqmats** option)
*DLOAD	**pr, pri**
*DRUCKER PRAGER	**aqpddm** (**abaqmats** option)
*DYNAMIC	**dynamic** (**abstep** option)
*EL FILE	**ersltf** (**abstep** option)
*EL PRINT	**edataf** (**abstep** option)
*ELASTIC	**aqelas** (**abaqmats** option)
*ELEMENT	**c3d, c3dh, c3di, c3dih, c3dr, c3drh, c3dm, c3dmh, c3dt, c3dht, c3drt, c3drht, c3dmt, c3dmht, dc3d, dcc3d, dcc3dd, dc3de, c3dp, c3dph, c3drp, c3drph, c3dmp, c3dmph, ac3d, c3de, c3dre, m3d, m3dr, s, sr, sr5, ds, srt, t3d, t3dh, t3dt, t3de, b3, b3h, pipe3, pipe3h, b3os, b3osh** (**abaqmats** option)
*ELEMENT OUTPUT	**fielo, histo** (**abstep** options)
*ELSET	**eset, eseti**
*END STEP	**abstep**
*ENERGY FILE	**ersltf** (**abstep** option)
*ENERGY OUTPUT	**fielo, histo** (**abstep** options)
*ENERGY PRINT	**endataf** (**abstep** option)
*EQUATION	**mpc, jd, jt**
*EXPANSION	**aqeps** (**abaqmats** option)
*FAILURE RATIOS	**aqfara** (**abaqmats** option)
*FREQUENCY	**frequency** (**abstep** option)
*FRICTION	**sid**
*GEOSTATIC	**geostati** (**abstep** option)
*HEADING	**title**
*HEAT GENERATION	**aqheat** (**abaqmats** option)
*HEAT TRANSFER	**heat** (**abaqmats** option)
*HYPERELASTIC	**aqhyper** (**abaqmats** option)

*HYPOELASTIC	**aqhypo** (**abaqmats** option)
*INELASIC HEAT FRACTION	**aqinelst** (**abaqmats** option)
*INITIAL CONDITIONS	**ve,vei, velocity, rotation**
*LATENT HEAT	**aqlath** (**abaqmats** option)
*MEMBRANE SECTION	**aqeltyp** (**abaqmats** option)
*MASS	**pm, npm**
*MATERIAL	**aqabmats**
*MODAL DYNAMIC	**moddyn** (**abstep** option)
*MODAL FILE	**mrsltf** (**abstep** option)
*MODAL PRINT	**mdataf** (**abstep** option)
*MODAL OUTPUT	**fielo, histo** (**abstep** options)
*NO COMPRESSION	**aqnocs** (**abaqmats** option)
*NO TENSION	**aqnots** (**abaqmats** option)
*NODE	**block, cylinder, bm, jt, spring, npm**
*NODE FILE	**nrsltf** (**abstep** option)
*NODE OUTPUT	**fielo, histo** (**abstep** options)
*NODE PRINT	**ndataf** (**abstep** option)
*NSET	**nset, nseti**
*ORIENTATION	**aqorient** (**abaqmats** option)
*ORNL	**aqornl** (**abaqmats** option)
*OUTPUT	**fielo, histo** (**abstep** options)
*PERMEABILITY	**aqperm** (**abaqmats** option)
*PLASTIC	**aqplas** (**abaqmats** option)
*POROUS BULK MODULI	**aqpbmptr** (**abaqmats** option)
*POROUS ELASTIC	**aqpore** (**abaqmats** option)
*POTENTIAL	**aqayld** (**abaqmats** option)
*RADIATION FILE	**rdataf** (**abstep** option)
*RADIATION OUTPUT	**fielo, histo** (**abstep** options)
*RADIATION PRINT	**rrsltf** (**abstep** option)
*RANDOM RESPONSE	**random** (**abstep** option)
*RATE DEPENDENT	**aqrdvp** (**abaqmats** option)
*RATIOS	**aqanswel** (**abaqmats** option)
*RESPONSE SPECTRUM	**response** (**abstep** option)
*SECTION CONTROLS	**scontrol, hourglas, kinsplit, sorder, weight, aqorient** (**abaqmats** options)
*SECTION FILE	**srsltf** (**abstep** option)
*SECTION PRINT	**sdataf** (**abstep** option)
*SHEAR RETENTION	**aqsret** (**abaqmats** option)
*SHELL GENERAL SECTION	**aqeltyp** (**abaqmats** option)
*SOILS	**soils** (**abstep** option)
*SOLID SECTION	**aqeltyp** (**abaqmats** option)
*SPECIFIC HEAT	

*STATIC	**static** (**abstep** otion)
*STEADY STATE DYNAMICS	**ssdyn** (**abstep** option)
*STEP	**abstep**, and **amplitud, cycle, inc, monitoni, nlgeom, rottol, submax** (**abstep** options)
*SURFACE	**si, sii** (see also **sid**)
*SURFACE INTERACTION	**sid**
*SWELLING	**aqswel** (**abaqmats** option)
*TENSION STIFFENING	**aqtens** (**abaqmats** option)
*TRANSFORM	
*USER MATERIAL	**aqusmt** (**abaqmats** option)
*VISCO	**visco** (**abstep** option)
*VISCOELASTIC	**aqvisco** (**abaqmats** option)

In case a "*" keyword command is not generated by **True*Grid***®, use the **verbatim** command to create the exact line to be replicated in the output file. You may also wish to contact XYZ Scientific Applications at (925) 373-0628 or at info@truegrid.com to request that this feature be supported in later versions of **True*Grid***®. The **verbatim** command saves you from inserting the "*" keyword command into the ABAQUS® input deck. This is particularly useful if you are rerunning the **True*Grid***® session file as you evolve the model or make parametric changes to it.

Steps

Use the **abaqstep** command to define a single step in the analysis. You can select the type of analysis, associated parameters, loads, and output. Details are found below.

Sliding (or Contact) Surfaces

To form a contact surface, use the **sid** command to define the surface type. Others are formed partially from nodes. The **sid** command also has optional parameters such as friction.

While in the part phase use the **si** or **sii** commands to select faces of that part for inclusion in the surface definition. If the face is from a shell element, be sure to use the **orpt** orientation command prior to issuing the **si** or **sii** command so that the orientation of the face is towards the opposing face in the sliding surface. If you are using part replication (**lrep**, **grep**, or **pslv**), then you may want to use the **lsii** or the **gsii** to increment the sliding interface command for each replication. You must use the **sid** command for each sliding surface that is referenced when the **lsii** or **gsii** commands are used with replication.

You can use sets in the merge phase to add faces or nodes to a sliding surface. These sets can be formed with the combined use of the **fset** (for faces) and the **nset** (for nodes) commands in the part and merge phase. Only use node sets when defining a sliding surface where nodes are on the slave

side and otherwise only use face sets. The node density between the master and slave sides of the interface should be roughly equal. When forming the mesh in the part phase, it may be necessary to build into the mesh a small gap between the master and slave sides of the contact surfaces, depending on the mesh density and the curvature to avoid initial penetration of the slave side into the master side.

When you merge the nodes (in the merge phase), the nodes from the slave side will not be allowed to merge with the nodes on the master side. Use the **mns** command in the merge phase to override this condition. When you fist merge the nodes, a table will be printed to the text window and the tsave file listing the number of faces and nodes associated with each sliding surface. Check this table carefully. You can also see the faces and nodes of either side of the sliding surfaces using the **co** command. When using this in combination with the hide graphics option, you can see the orientation of the faces. Use **labels** command to show how the nodes have merged graphically.

Initial and Boundary Conditions

There are several ways to constrain nodes. The **b** and **bi** commands in the part phase or the **b** command in the merge phase will constrain nodes in the global coordinate system. Use the **plane** command to specify symmetry plane constraints including symmetry planes with failure. Nodes in the model will be assigned to these symmetry planes based on the tolerance you specify in the **plane** command. The **lb** (and the associated **lsys**) command can be used to set the constraints in any coordinate system. The **sfb** command can also be used to do this. Be sure that something in the model has been constrained or the entire model might fly off.

If you use the **velocity** or **rotation** command in the control phase, then all subsequent parts will be assigned this initial velocity. This can be over ridden using the **velocity** or **rotation** command within a part. Both of these conditions can be over ridden for specific regions of the mesh using the **ve** or **vei** commands in the part phase or the **ve** command in the merge phase. Velocities are not accumulative. Care is needed when assigning initial velocities so that when two nodes are merged, the velocities of those two nodes match. Only one of the velocities will be used and if they do not match, you may get an unexpect result. Usually, if the velocities of two merged nodes do not match, this indicates an error in the model.

Loads

There are numerous ways to assign loads. Every command that generates a load has a load curve or set id number associated with it. This number is used in the **abaqstep** command to tie each load to a step. The list of commands that can be used to assign loads in the part phase includes:

fc	Cartesian concentrated nodal loads
fci	Cartesian concentrated nodal loads
fcc	cylindrical concentrated nodal loads

fcci	cylindrical concentrated nodal loads
fcs	spherical concentrated nodal loads
fcsi	spherical concentrated nodal loads
mom	nodal moment about one of the nodal axis in the global coordinate system
momi	nodal moment about one of the nodal axis in the global coordinate system
ndl	pressure converted to distributed nodal loads
ndli	pressure converted to distributed nodal loads
pr	pressure loads on element faces
pri	pressure loads on element faces
pramp	pressure loads on element faces
fv	Cartesian prescribed nodal velocities
fvi	Cartesian prescribed nodal velocities
fvc	cylindrical prescribed nodal velocities
fvci	cylindrical prescribed nodal velocities
fvs	spherical prescribed nodal velocities
fvsi	spherical prescribed nodal velocities
fvv	Cartesian variable prescribed nodal velocities
fvvi	Cartesian variable prescribed nodal velocities
fvvc	cylindrical variable prescribed nodal velocities
fvvci	cylindrical variable prescribed nodal velocities
fvvs	spherical variable prescribed nodal velocities
fvvsi	spherical variable prescribed nodal velocities
acc	Cartesian prescribed nodal acceleration
acci	Cartesian prescribed nodal acceleration
accc	cylindrical prescribed nodal acceleration
accci	cylindrical prescribed nodal acceleration
accs	spherical prescribed nodal acceleration
accsi	spherical prescribed nodal acceleration
vacc	Cartesian variable prescribed nodal acceleration
vacci	Cartesian variable prescribed nodal acceleration
vaccc	cylindrical variable prescribed nodal acceleration
vaccci	cylindrical variable prescribed nodal acceleration
vaccs	spherical variable prescribed nodal acceleration
vaccsi	spherical variable prescribed nodal acceleration
fd	Cartesian displacement
fdi	Cartesian displacement
fdc	cylindrical displacement
fdci	cylindrical displacement
fds	spherical displacement
fdsi	spherical displacement

The list of commands that can be used to assign loads in the merge phase includes:

fc	Cartesian concentrated nodal loads
mom	nodal moment about one of the nodal axis in the global coordinate system
ndl	pressure converted to distributed nodal loads
pr	pressure loads on element faces
pramp	pressure loads on element faces
fv	Cartesian prescribed nodal velocities
fvv	Cartesian variable prescribed nodal velocities
vacc	Cartesian variable prescribed nodal acceleration
fd	Cartesian displacement

The **pramp** command is used with either **pr** or **pri**. It applies a pressure based on a function for all nodes that have a zero pressure. In most cases, the magnitude of the load is specified using a load curve. This varies the amplitude of the load with respect to time.

Load Curves

Load curves are 2D polygonal curves that can be created using the **lcd** and **flcd** commands. Load curves are typically used to define the relative amplitude of a load with respect to time. They can be used to relate any two variables. Almost all prescribed loads require a load curve in time so that the amplitude of the load can vary. It is best to define a load curve before it is referenced in a load or material model to avoid a warning message.

In some dialogue boxes you might be prompted for a load curve or a set id. This is because such commands can be used to define, for example, a dynamic load or a static load that has the option to turn loads on or off in the **abstep** command.

Bricks

Brick elements refer to hexahedral, prism (wedge), and tetrahedral elements. Most, but not all, materials support the different brick element types. There are no section properties for bricks. Be sure to use the **mate**, **mt**, or **mti** command to assign the proper material to each section of the mesh.

The element local coordinate system used in an orthotropic or anisotropic material is imposed by the order of the nodes that define the element. You can flip the nodal ordering to switch the orientation of this local coordinate system using the **or** command in the part phase.

Shells

Shell elements refer to both quadrilateral and triangular elements and sometimes referred to as structural elements. Cross sectional properties are included in the material model when the shell type

is selected. There are no section properties for bricks. Be sure to use the **mate**, **mt**, or **mti** command to assign the proper material to each section of the mesh. The default shell thicknesses are included as part of the cross sectional properties. These default thicknesses can be over ridden with the use of the **thic** command in the part phase. Both can be over ridden for a region of the part using the **th** and **thi** commands. If you have two surfaces that represent the inner and outer surfaces of a structure that is to be modeled using shell elements, than you can use the **ssf** and **ssfi** commands in the part phase to create shells with variable thickness.

The orientation of the positive normal direction to the shell is dictated by the nodal ordering of the nodes that define the shell. This positive direction is used, for example, to determine the direction of a positive pressure. This direction can be flipped using the **n** command in the part phase. The order of the nodes also dictate the local material coordinate system which can be important when using an orthotropic or anisotropic material. Use the **or** command to flip the coordinate system to the desired direction. When an angle is specified for the orientation of a composite material, it is with respect to this orientation.

Beams

Two nodes are required to form a beam element. In many cases, a third node is needed to define the local coordinate system used to form the cross sectional properties. These element are sometimes referred to as structural elements. Use the **ibm**, **ibmi**, **jbm**, **jbmi**, **kbm**, and **kbmi** commands to form beam elements with shell or brick structures while in the part phase. If the material of the shell or brick structure is set to zero using the **mt**, **mti**, or **mate** command, then the shells or bricks will be ignored, but the embedded beams will not be ignored. This is a convenient way to build an array of beams using block structured methods. You can also use the **bm** command in the merge phase to build a string of beams that can be made to follow a 3D curve. The **beam** command (this command has been denigrated) can also be used to form beam elements, but the command is not interactive.

Both the element type and the default cross section properties are defined in the material definition. You can also use the **bsd** command to define cross sectional properties to over ride the material default cross sectional properties. When you create a beam, refer to the **bsd** number to assign these cross sectional properties to the beam.

Point Masses

Point masses can be generated in the part or merge phase. There are two types of point masses. The **pm** command will assign a mass to an existing node. The **npm** will create a new node and assign it a mass. The latter must then be connected either to a spring or beam.

Shared Constraints

Use the **mpc** command to couple a set of nodes. This requires that you create a node set first. The **nset** or **nseti** command can be used in the part phase and the **nset** command in the merge phase to create a node set. Also, click on the pick button in the environment window during the merge phase. Then you can use the mouse to modify or create a node set. The nodes sharing a set of constraints will not be merged together.

Post Processing

The **abaqstep** command is used to identify the variables that are to be processed for post processing. There are four post processing options: standard results files, data file requests, field output, and history output. Standard results files are specified with the **abastep** options **crsltf**, **ersltf**, **nrsltf**, **mrsltf**, **rrsltf**, **srsltf**, **enrsltf**. These can be repeated as many times as is needed. There may be some optional arguments followed by a list of variables. This list of variables are ended with a semicolon. The dialogue box helps make it easy to select the variables from a list, but the list is quite long. For data file requests, use the options **cdataf**, **edataf**, **ndataf**, **mdataf**, **rdataf**, **sdataf**, **endataf**. Field and history variables are specified using the **fielo** and **histo** options.

II. ABAQUS® Example

III. ABAQUS® Output Reference

The syntax for commands are described below were literals are highlighted in bold. Symbols to be substituted are italicized. Many options in these commands corresponds to an ABAQUS keyword command. This is frequently noted in the text below with the "*KEYWORD" in the right column of the description of the option. Each command is described by an entry like the following:

Command Syntax Conventions

When an arbitrarily long list of arguments are required, a semi-colon terminates the list. When a semi-colon is found in the description of an option or command, this indicates such a list. It is common to have a list inside another list. Each list must have a terminating semi-colon. This is analogous to parenthesis in algebraic expressions where the opening parenthesis must be balanced with a closing parenthesis. In this case, the keyword that initiates a list of items must be balanced with a closing semi-colon. Sometimes a short list of arguments and options can be repeated indefinitely, forming a list. The set of arguments and options that can be repeated are placed in square brackets. Sometimes the abbreviation *#_things* is used to mean "number of things". Each command is described by an entry like the following:

command summary description

command *arguments* brief description of functionality
with brief descriptions of what the *arguments* should be.
indentation is used to indicate a list of options to the *arguments*

Some commands in the part phase require a region specification. The region selects a face of the mesh, among other things. Others may require a progression specification. The progression selects multiple faces, among other things. In the merge phase, such commands require an option. In all of these cases, a portion of the mesh is identified. For example, the **si**/**sii** command has this property.

Remarks

When present, the Remarks section describes the command in even greater detail. It may describe the context in which the command is normally used, and other commands used in association with this command. It may describe side effects. It may describe other, similar commands. In many cases, it includes a description of where to find the command in the menus.

Examples

When present, this shows the exact use of the command. If you use the dialogues, this command will be generated by simple selection options with the mouse and entering data where indicated. The

command, as shown here, will appear in the session file for later reuse and possible modification. You can also enter the command into the text window or insert it into a command file to be run in batch mode.

abaqmats ABAQUS® materials

abaqmats *material_# options* ;
where an *option* can be any of the following:
aqeltyp *type* select the family of element types
where *type* can be

c3d	standard solid stress/displacement
c3dh	hybrid solid stress/displacement
c3di	incompatible modes solid stress/displacement
c3dih	incompatible modes hybrid solid stress/displacement
c3dr	reduced integration solid stress/displacement
c3drh	reduced integration hybrid solid stress/displacement
c3dm	modified solid stress/displacement
c3dmh	modified hybrid solid stress/displacement
c3dt	standard solid coupled temperature displacement
c3dht	hybrid solid coupled temperature displacement
c3drt	reduced integration solid coupled temperature displacement
c3drht	reduced integration hybrid solid coupled temperature displacement
c3dmt	modified solid coupled temperature displacement
c3dmht	modified hybrid solid coupled temperature displacement
dc3d	solid diffusive heat/mass diffusion
dcc3d	standard solid forced convection/diffusion
dcc3dd	w/ dispersion control solid forced convection/diffusion
dc3de	solid coupled thermal-electric
c3dp	standard solid pore pressure
c3dph	hybrid solid pore pressure
c3drp	reduced integration solid pore pressure
c3drph	reduced integration hybrid solid pore pressure
c3dmp	modified solid pore pressure
c3dmph	modified hybrid solid pore pressure
ac3d	solid acoustic
c3de	solid piezoelectric
c3dre	reduced integration solid piezoelectric
m3d	standard membrane
m3dr	reduced integration membrane
s	standard shell stress/displacement
sr	reduced integration shell stress/displacement

sr5 reduced integration w/ 5 dofs shell stress/displacement
ds shell heat transfer
srt shell coupled temperature-displacement
t3d standard truss stress/displacement
t3dh hybrid truss stress/displacement
t3dt truss coupled temperature displacement
t3de truss piezoelectric
b3 standard beam
b3h hybrid beam
pipe3 pipe
pipe3h pipe hybrid
b3os open section beam
b3osh hybrid open section beam

aqdens *density* mass density *DENSITY
aqdepv *#_variables* number of dependent variables
aqtherm *opt1* thermal expansion coefficients
where *opt1* can be any of the following:
aqexze *temp_0* initial temperature
aqnpm *opt2* non-porous material
where *opt2* must be one of the following:
aqexis [*alpha tmpopts* ;] ; isotropic
where a *tmpopt* can be
aqotmp *temperature*
aqofv *variable_list* ; variable dependencies
aqexor [*alpha_11 ... alpha_33 tmpopt fldopt*] ; orthotropic
aqexan [*alpha_11 ... alpha_23 tmpopt fldopt*] ; anisotropic
aqmstr *opt2* material structural
where *opt2* must be one of the following:
aqexis [*alpha tmpopt stropt*] ; isotropic
aqexor [*alpha_11 ... alpha_33 stropt tmpopt*] ; orthotropic
aqexan [*alpha_11 ... alpha_23 stropt tmpopt*] ; anisotropic
aqporf [*alpha tmpopt fldopt*] ; pore fluid
aqcond *opt1* conductivity
where *opt1* must be one of the following:
aqcdis [*conductivity tmpopt fldopt*] ; isotropic
aqcdor [*k_11 k_22 k_33 tmpopt fldopt*] ; orthotropic
aqcdan [*k_11 ... k_33 tmpopt fldopt*] ; anisotropic
aqlath [*heat temp temp*] ; latent heat
aqspec [*specific_heat tmpopt fldopt*] ; specific heat
aqperm *opt1* permeability
where *opt1* can be any of the following:

aqsww *weight* specific weight of water

aqpris *k* isotropic

aqpror *k11 k22 k33* orthotropic

aqpran *k11 ... k33* anisotropic

aqvoid voids ratio

aqporo [*temp opt1*] ; porous bulk moduli

where *opt1* can be any of the following:

aqslgr *modulus* solid grain bulk modulus

aqprfl *modulus* permeating fluid bulk modulus

aqacm *modulus* [*drag frequency*] ; acoustic medium

aqdepl [*modulus ratio yield exp offset temp*] ; deformation plasticity

aqusmt *opt1 list_params* ; user material

where *opt1* must be one of the following:

aqumsy symmetric

aqumusy unsymmetric

aqelas *opt1* ;

where *opt1* must be one of the following:

aqelis [*modulus ratio tmpopt fldopt*] ; isotropic

aqelec [*e1 e2 e3 v12 v13 v23 g12 g13 g23 tmpopt fldopt*] ; engineering

aqella [*e1 e2 v12 g12 g13 g23 tmpopt fldopt*] ; lamina

aqelor [*d1111 ... d2323 tmpopt fldopt*] ; orthotropic

aqelan [*d1111 ... d2323 tmpopt fldopt*] ; anisotropic

aqhyper *opt1* - hyperelastic

where *opt1* must be one of the following:

aqseps strain energy by user subroutine

aqsepp 1 [*c10 c01 d1 r tmpopt fldopt*] ; order 1

aqsepp 2 [*c10 ... c02 d1 d2 r tmpopt fldopt*] ; order 2

aqsepp 3 [*c10 ... c03 d1 d2 d3 r tmpopt fldopt*] ; order 3

aqsepp 4 [*c10 ... c04 d1 d2 d3 d4 r tmpopt fldopt*] ; order 4

aqhypo [*modulus ratio i1 i2 i3*] ; hypoelastic

aqpore *opt1* porous elastic

where *opt1* must be one of the following:

aqctsm [*bulk shear limit tmpopt fldopt*] ; constant shear

aqcpsm [*bulk ratio limit tmpopt fldopt*] ; Poisson's

aqheat heat generation

aqmpmf *factor* mass proportional damping factor

aqspdf *factor* stiffness proportional damping factor

aqfcdf *factor* fraction for composite damping factor
aqnocs allow no compression stress
aqnots allow no tension stress
aqconc [stress strain] ; concrete
aqtens [stress strain] ; tension stiffening
aqfara *opt1* ; failure ratios
where *opt1* can be any of the following:
aqfr1 *ratio* ultimate biaxial/uniaxial compression stress
aqfr2 *ratio* uniaxial tension/compression stress at failure
aqfr3 *ratio* plastic strain at ultimate stress biaxial/uniaxial
aqfr4 *ratio* principle stress/uniaxial stress at cracking
aqsret *opt1* ; shear retention
where *opt1* can be any of the following:
aqshrt1 *rho* dry concrete
aqshrt2 *epsilon* dry concrete
aqshrt3 *rho* wet concrete
aqshrt4 *epsilon* wet concrete
aqdppm [*list_opt1 tmpopt fldopt* ;] ; Drucker Prager plasticity
where *opt1* can be any of the following:
aqdpm1 *angle* material angle of friction
aqdpm2 *k* ratio of flow stress in triaxial tension/compression
aqdpm3 *angle* dilation angle
aqplas *opt1* [*stress strain tmpopt fldopt*] ; plastic
where *opt1* can be one of the following:
aqishrd isotropic hardening
aqkihard kinematic hardening
aqayld *sigma11 sigma22 sigma33 tau12 tau13 tau23* potential
aqrdvp [*d p tmpopt fldopt*] ; rate dependent
aqcree *opt1* - creep
where *opt1* must be one of the following:
aquscre user subroutine
aqthcre [*a n m temp*] ; time hardening
aqshcre [*a n m temp*] ; strain hardening
aqhscre [*a b n dh r*] ; hyperbolic sine
aqornl *opt1* ornl
where *opt1* can be any of the following:
aqaornl *rate* saturation rate for kinematic shift
aqhornl *rate* rate of kinematic shift w.r.t. creep strain
aqmornl stainless steel hardening
aqrornl invoke optional alpha reset procedure

aqswel *opt1* swelling
where *opt1* must be one of the following:
aqusswe user subroutine
aqdtswe [*strain_rate tmpopt fldopt*] ; data specified
aqanswel *r11 r22 r33* ratios
aqclay clay plasticity
aqint *modulus stress_ratio beta k* specify intercept
aqnoint *modulus stress_ratio surf_size beta k* no intercept
aqcycl *opt1* [*stress tmpopt fldopt;*] cycled plastic
where *opt1* can be one of the following:
aqcycl1 every 10th cycle
aqcycl2 every 100th cycle
aqinelst *heat_flux* inelastic heat fraction
aqvisco *opt1* viscoelastic
where *opt1* can be one of the following:
aqvisc1 *real_g1 imag_g1 a real_k1 imag_k1 b* formula
aqvisc2 [*real_wg imag_wg real_wk imag_wk frequency*] ; tabular
aqvisc3 [*opt2 time*] ; prony
where *opt1* can be any of the following:
aqprny1 *ratio* shear relaxation modulus ratio
aqprny2 *ratio* bulk relaxation modulus ratio
scontrol *factor1 factor2 factor3* section scale factors
hourglas *option*
where the *option* can be
enhanced enhanced
relax relax stiffness
stiffness stiffness
viscous viscous
kinsplit *option*
where the *option* can be
centroid centroid
orthogon orthogonal
sorder select second order
weight *wf* weight factor
aqorient *type args* assign an orientation to the elements
where the *type* and *args* can be
coor *x1,y1,z1,x2 y2 z2 ld angle* for coordinates

Remarks

This command is used in conjunction with the **mt**, **mti**, **mate**, and **mtv** commands. This command sets the global properties of a material model and identifies this model with a number. Then the **mt**,

mti, **mate**, and **mtv** commands can be used to associate elements with this model by its identification number.

When you select a family of element types, using the **aqeltyp** option, the appropriate element type will be made depending on the shape (hexahedron, prism, tetrahedron, quad shell, or triangle) of the element that you generate.

tmpopt means optionally "aqotmp temperature".
fldopt means optionally "aqofv list_field_values ;".
stropt means optionally "aqeps effective_stress".
short comments are added after first dash in a line.

abaqstep ABAQUS® analysis step

abaqstep *step_# procedure options* ; creates *STEP
where the *procedure* can be one of the following:
buckle *parameter* ; creates *BUCKLE
where the *parameter* must be one of:
dead dead loading
live live loading
ctd *parameters features* ; creates *COUPLED TEMPERATURE-DISPLACEMENT
where the ordered *parameters* that must follow are:
btol basic tolerance
ttol temperature tolerance
ststep suggested initial time step
totstep total time period for the step
where the unordered optional *features* are:
explicit explicit integration
mtol *tolerance* moments tolerance
nocreep no creep
steady steady state analysis
cetol *tolerance* creep tolerance
deltmx *temperature* maximum temperature change
cetol *tolerance* creep tolerance
deltmx *tolerance* maximum temperature change
timemin *time* minimum time increment
timemax *time* maximum time increment
dynamic *parameters features* ; creates *DYNAMIC
where the ordered *parameters* must be:
method

which can be one of the following:

explicit	explicit integration
subspace	subspace projection method
implicit *tolerance*	implicit integration
timeinc	suggested time increment
time	time period

where the unordered optional *features* can be:

direct	user control of step size (explicit only)
vectors *#_modes*	set the number of modes (subspace only)
alpha *alpha*	artificial damping control
haftol *tolerance*	half-step residual tolerance
initial	no initial accelerations
nohaf	no half-step residual
mtol *tolerance*	moments tolerance
timemin *time*	minimum time increment
timemax *time*	maximum time increment

frequenc *parameters features* ; creates *FREQUENCY

where the ordered *parameters* must be:

ne	number of eigenvalues
maxfreq	maximum frequency

where the unordered optional *features* can be:

shift *frequency_squared*	shift point
nvecs *n*	number of vectors
maxit *n*	number of iterations

geostati *parameter feature* ; creates *GEOSTATIC

where the *parameter* must be:

*tol*erance	tolerance

where the optional *feature* is:

mtol *tolerance*	moments tolerance

heat *parameters features* ; creates *HEAT TRANSFER

where the ordered *parameters* must be:

temp	temperature tolerance
times	time step
timep	time period
timeinc	minimum time increment

where the unordered optional *features* can be:

deltmx *temp*	maximum temperature change
endcon *ss*	steady state ending condition
endcon *period*	periodic
steady	steady state analysis
timmxinc *time*	maximum time increment
temprate *time*	temperature change rate

moddyn *parameter features* ; creates *MODAL DYNAMIC
where the *parameter* must be:
time for total time
where the unordered optional *features* can be:
initial yes start new dynamic response
initial no use last dynamic response
random *parameters* ; creates *RANDOM RESPONSE
where the ordered *parameters* must be:
lfreq lowest frequency
hfreq highest frequency
n number of points
bias bias
i frequency scale
response *parameters* ; creates *RESPONSE SPECTRUM
where the ordered *parameters* can be repeated up to 2 times:
name name of response spectrum
x x-direction cosine
y y-direction cosine
z z-direction cosine
scale magnitude
soils *parameters features* ; creates *SOILS
where the ordered *parameters* must be:
tol tolerance
times initial time step
timep total time period for the step
where the unordered optional *features* can be:
consolid transient, consolidated analysis
endcon *ss* end at steady state
endcon *period* periodic
mtol *tol* moments tolerance
utol *tol* maximum pore pressure change
tmmninc *time* minimum time increment
tmmxinc *time* maximum time increment
presrate *rate* minimum pore pressure rate of change
static *parameters features* ; creates *STATIC
where the ordered *parameters* must be:
tol tolerance
times initial time step
timep total time period for the step
where the unordered optional *features* can be:
mtol *tol* moments tolerance

tmmninc *time* minimum time increment
tmmxinc *time* maximum time increment

ssdyn *parameters features* ; creates *STEADY STATE DYNAMIC

where the ordered *parameters* must be:

freq lowest frequency
freq highest frequency
n number of points
bias bias
i frequency scale

visco *parameters features* ; creates *VISCO

where the ordered *parameters* must be:

tol tolerance
times suggested initial time step
timep total time period for the step

where the unordered optional *features* can be:

cetol *tol* maximum creep strain rate
explicit explicit integration
mtol *tol* moments tolerance
tmmninc *time* minimum time increment
tmmxinc *time* maximum time increment

where the *procedure* definition is followed by unordered *options* which can be:

amplitude *flag*

where *flag* can be

step stepped amplitude
ramp ramped amplitude

cycle #_*inter* maximum iterations in an increment
inc #_*inter* maximum increments in a step
linear new linear analysis with a new stiffness matrix
linear old linear analysis with old stiffness matrix
monotoni monotonic
nlgeom geometric non-linearity
rottol *tol* maximum increment of rotation
submax suppress subdivisions
[**abdload blc** *load_curve_# type*] associated distributed loads *DLOAD

where *type* can be

pr pressure

[**abcload blc** *load_curve_# type*] associated concentrated loads *CLOAD

where *type* can be

fc concentrated force
mom concentrated moments
fd displacement
fv velocity

acc acceleration
ft forced temperature

[**crsltf** *options keys* ;] contact results file *CONTACT FILE

where an *option* can be

freq *frequency* frequency
sinm *face_set* master sliding interface
sins *face_set* slave sliding interface
namens *set_name* name of node set

where *keys* is a space delimited list of variable names
(see the ABAQUS User's Manual for the complete list)

[**ersltf** *options keys* ;] element results file *EL FILE

where an *option* can be

dirw directions
elesn *set_name* element set name
freq *frequency* frequency
lmode *mode* last mode
fmode *mode* first mode
posi *flag* position

where *flag* can be

1 averaged at nodes
2 centroidal
3 integration points
4 nodes

reba *name* rebar

where *keys* is a space delimited list of variable names
(see the ABAQUS User's Manual for the complete list)

[**nrsltf** *options keys* ;] nodal results file *NODE FILE

where an *option* can be

freq *frequency* frequency
noglob no global directions
lmode *mode* last mode
fmode *mode* first mode
namens *set_name* node set name

where *keys* is a space delimited list of variable names
(see the ABAQUS User's Manual for the complete list)

[**mrsltf** *options keys* ;] modal results file *MODAL FILE

where an *option* can be

freq *frequency* frequency

where *keys* is a space delimited list of variable names
(see the ABAQUS User's Manual for the complete list)

[**rrsltf** *options keys* ;] radiation results file *RADIATION FILE

where an *option* can be

freq *frequency*	frequency
cavi *cavity_name*	cavity
namees *set_name*	element set
surf *surface_name*	surface

where *keys* is a space delimited list of variable names
(see the ABAQUS User's Manual for the complete list)

[**srsltf** *surface section options keys* ;] section results file *SECTION FILE

where an *option* can be

laxe	local axes output
freq *frequency*	frequency
nupd	no update
dann *node_#*	anchor node
danc *x y z*	anchor point
daxn1 *node_#*	first axis node
daxc1 *x y z*	first axis point
daxn2 *node_#*	second axis node
daxc2 *x y z*	second axis point

where *keys* is a space delimited list of variable names
(see the ABAQUS User's Manual for the complete list)

[**enrsltf** *options* ;] energy results file *ENERGY FILE

where an *option* can be

namees *set_name*	element set name
freq *frequency*	frequency

where *keys* is a space delimited list of variable names
(see the ABAQUS User's Manual for the complete list)

[**cdataf** *options keys* ;] contact data file *CONTACT PRINT

where an *option* can be

freq *frequency*	frequency
sin *interface_#*	sliding interface number
namens *set_name*	node set name
nsum	no summary
tota	totals

where *keys* is a space delimited list of variable names
(see the ABAQUS User's Manual for the complete list)

[**edataf** *options keys* ;] element data file *EL PRINT

where an *option* can be

elesn *set_name*	element set name
freq *frequency*	frequency
lmode *mode*	last mode
fmode *mode*	first mode
posi *flag*	position

where *flag* can be

1	averaged at nodes
2	centroidal
3	integration points
4	nodes

reba *name*	rebar
nsum	no summary
tota	totals

where *keys* is a space delimited list of variable names
(see the ABAQUS User's Manual for the complete list)

[**ndataf** *options keys* ;]	nodal data file	*NODE PRINT

where an *option* can be

freq *frequency*	frequency
glob	global
jmode *mode*	last mode
fmode *mode*	first mode
namens *set_name*	node set name
nsum	no summary
tota	totals

where *keys* is a space delimited list of variable names
(see the ABAQUS User's Manual for the complete list)

[**mdataf** *options keys* ;]	modal data file	*MODAL PRINT

where an *option* can be

freq *frequency*	frequency

where *keys* is a space delimited list of variable names
(see the ABAQUS User's Manual for the complete list)

[**rdataf** *options keys* ;]	radiation data file	*RADIATION PRINT

where an *option* can be

freq *frequency*	frequency
cavi *name*	cavity
namees *set_name*	element set
surf *name*	surface
nsum	no summary
tota	totals

where *keys* is a space delimited list of variable names
(see the ABAQUS User's Manual for the complete list) *SECTION PRINT

[**sdataf** *surface section options keys* ;] section data file

where an *option* can be

cavi *name*	cavity
namees *set_name*	element set
surf *name*	surface

daxn $node_1$ $node_2$ define axes by nodes
daxc x_1 y_1 z_1 x_2 y_2 z_2 define axes by coordinates
where *keys* is a space delimited list of variable names
(see the ABAQUS User's Manual for the complete list)

[**endataf** *options* ;] energy data file *ENERGY PRINT
where an *option* can be
namees *set_name* element set name
freq *frequency* frequency

[**fielo** *options vars*] output field *OUTPUT, FIELD
where an *option* can be
oni *n* number of intervals
tim *flag* time marks
where *flag* can be
yes
no
cnew start from scratch
cadd add to previous options
crepl replace only similar types
where *vars* must one of
all all variables
list *lists* ;
where a *list* can be
cont *options list* ; contact variables *CONTACT OUTPUT
where an *option* can be
cpset *set_name* contact pair
contact contact
nset *set_name* node set
master *surface_name* master side
slave *surface_name* slave side
where a *list* must be one of:
all all energy variables
preselec preselected variables
keys list of variables
where *keys* is a space delimited list of variable names
(see the ABAQUS User's Manual for the complete list)
elem *options list* ; element variables *ELEMENT OUTPUT
where an *option* can be
elset *set_name* element set
position *loc* position
where *loc* can be
cent centroidal

integ integration points
nodes nodes
rebar *name* rebar
where a *list* must be one of:
all all energy variables
preselec preselected variables
keys list of variables
where *keys* is a space delimited list of variable names
(see the ABAQUS User's Manual for the complete list)

node *options list* ; node variables * N O D A L OUTPUT
where an *option* can be
nset *set_name* node set
tracer *name* tracer
where a *list* must be one of:
all all energy variables
preselec preselected variables
keys list of variables
where *keys* is a space delimited list of variable names
(see the ABAQUS User's Manual for the complete list)

radi *options list* ; radiation variables *RADIATION OUTPUT
where an *option* can be
cavity *name* cavity
elset *set_name* element set
surface *surface_name* surface
where a *list* must be one of:
all all energy variables
preselec preselected variables
keys list of variables
where *keys* is a space delimited list of variable names
(see the ABAQUS User's Manual for the complete list)

[**histo** *options vars*] output history *OUTPUT, HISTORY
where an *option* can be
freq *interval* frequency interval
lmod *list_modes* ; mode list
where *vars* must one of
all all variables
list *lists* ;
where a *list* can be
cont *options list* ; contact variables *CONTACT OUT-PUT

where an *option* can be

cpset *set_name*	contact pair
nset *set_name*	node set
master *surface_name*	master side
slave *surface_name*	slave side

where a *list* must be one of:

all	all energy variables
preselec	preselected variables
keys	list of variables

where *keys* is a space delimited list of variable names
(see the ABAQUS User's Manual for the complete list)

elem *options list* ; element variables *ELEMENT OUTPUT

where an *option* can be

elset *set_name*	element set
tracer *set_name*	tracer
rebar *name*	rebar

where a *list* must be one of:

all	all energy variables
preselec	preselected variables
keys	list of variables

where *keys* is a space delimited list of variable names
(see the ABAQUS User's Manual for the complete list)

node *options list* ; node variables * N O D A L OUTPUT

where an *option* can be

nset *set_name*	node set
tracer *set_name*	tracer

where a *list* must be one of:

all	all energy variables
preselec	preselected variables
keys	list of variables

where *keys* is a space delimited list of variable names
(see the ABAQUS User's Manual for the complete list)

moda *list* ; modal variables *MODAL OUTPUT

where a *list* must be one of:

all	all energy variables
preselec	preselected variables
keys	list of variables

where *keys* is a space delimited list of variable names
(see the ABAQUS User's Manual for the complete list)

radi *options list* ; radiation variables *RADIATION OUTPUT

where an *option* can be

cavity *name* cavity
elset *set_name* element set
surface *surface_name* surface
where a *list* must be one of:
all all radiation variables
preselec preselected variables
keys list of variables
where *keys* is a space delimited list of variable names
(see the ABAQUS User's Manual for the complete list)

ener *option list* ; energy variables *ENERGY OUTPUT
where *option* can be
elset *set_name* element set
where a *list* must be one of:
all all energy variables
preselec preselected variables
keys list of variables
where *keys* is a space delimited list of variable names
(see the ABAQUS User's Manual for the complete list)

Remarks

This command is used to define each time/history step. It can be repeated as many times as needed. There are several components to each step. The step number and the procedure are required.

1. step number
2. procedure
3. procedural options
4. associated distributed loads
5. associated concentrated loads
6. data files
7. results files
8. field output options
9. history output options

Since this command is very complex, it is advised to use the dialogue box interactively to generate this command. The information in this manual is intended to help you if you need to modify this command once it was generated from the dialogue box and placed into the session file. Another good reason to use the dialogue box is that the variable names (*keys*) for the different data, results, and output options can be selected from a list. The variable names (*keys*) are not listed here since there are many of them and can be found in the ABAQUS User's Manual.

There are 13 procedures to select from. Each procedure has a set of required parameters and additional options.

buckle to create	*BUCKLE
ctd to create	*COUPLED TEMPERATURE-DISPLACEMENT
dynamic to create	*DYNAMIC
frequenc to create	*FREQUENCY
geostati to create	*GEOSTATIC
heat to create	*HEAT TRANSFER
moddyn to create	*MODAL DYNAMIC
random to create	*RANDOM RESPONSE
response to create	*RESPONSE SPECTRUM
soils to create	*SOILS
static to create	*STATIC
ssdyn to create	*STEADY STATE DYNAMIC
visco to create	*VISCO

The procedure definition is followed by options which are listed and characterized below:

Procedural options

amplitude for the amplitude type
cycle for the maximum iterations in an increment
inc for the maximum increments in a step
linear for linear analysis stiffness matrix formation type
monotoni for monotonic
nlgeom for geometric non-linearity
rottol *tol* for maximum increment of rotation
submax to suppress subdivisions

Associated distributed facial loads

abdload identifies which distributed loads are to be included for this step. Use the **pr** and **pri** commands to select faces and amplitudes. Use the **lcd** and **flcd** commands to define the amplitude curves. *DLOAD

Associated concentrated nodal loads

abcload identifies which nodal loads are to be included for this step. There are many ways to select nodes for loads. Use the **lcd** and **flcd** commands to define the amplitude curves. *CLOAD

Standard Results files (repeat any of these command as many times as is needed)

crsltf to control writing to the contact results file *CONTACT FILE
ersltf to control writing to the element results file *EL FILE
nrsltf to control writing to the nodal results file *NODE FILE
mrsltf to control writing to the modal results file *MODAL FILE
rrsltf to control writing to the radiation results file *RADIATION FILE
srsltf to control writing to the section results file *SECTION FILE
enrsltf to control writing to the energy results file *ENERGY FILE

Data files requests (repeat any of these command as many times as is needed)

cdataf to define data file requests for contact variables *CONTACT PRINT
edataf to define data file requests for element variables *EL PRINT
ndataf to define data file requests for nodal variables *NODE PRINT
mdataf to define data file requests for modal variables *MODAL PRINT
rdataf to define data file requests for radiation variables *RADIATION PRINT
sdataf to define data file requests for section variables *SECTION PRINT
endataf to define data file requests for energy variables *ENERGY PRINT

Field output options (repeat this command as many times as is needed)

fielo produces the *OUTPUT card for FIELD output options and the associated *CONTACT OUTPUT, *ELEMENT OUTPUT, *NODAL OUTPUT, and *RADIATION OUTPUT.

History output options (repeat this command as many times as is needed)

histo produces the *OUTPUT card for HISTORY output options and the associated *CONTACT OUTPUT, *ELEMENT OUTPUT, *NODAL OUTPUT, *MODAL OUTPUT, *RADIATION OUTPUT, and *ENERGY OUTPUT.

Keys is a list of variable names. The list of possible variable names in this list is different for each option and it can be large. Refer to the ABAQUS User's Manual for the appropriate list. **True*Grid*®** does not check this list for validity.

Examples

```
abaqstep            c initiate a list of ABAQUS analysis options
     abstep 1            c step identification number
     static .001 .01     c static analysis
          mtol .0001          c timing instructions
          tmmninc .0001
          tmmxinc .0025
          ;                   c terminate static options list
     amplitude ramp      c method of applying loads
     cycle 10            c iteration controls
     inc 500
```

```
linear new          c stiffness matrix formation method
monotoni            c monotonic flag
nlgeom              c non linearity flag
cdataf              c contact print
     freq 2         c contact print parameters
     sinm msface1
     namens nodeset1
     sins ssface1
     nsum tota
     cstress cdisp sdv pfl ptl dbsf  ; c variable names
cdataf              c contact print
     freq 4         c contact print parameters
     hfl  ;         c variable names
histo               c history output
     freq 2
     cnew
     list elem      c elements output
     elset nn1
     elen elcd nforc  ;  c variable names
abdload blc 2 pr    c activate pressure - load curve 2
abcload blc 3 fc    c activate forces - load curve 3
;                   c terminate the ABAQUS analysis list
```

bm create a string of beam elements (merge phase)

bm *options ;*

where

option can be:

(Selection of the first node)

n1 *node_#*	to make an existing node the first node of the beams.
pm1 *point_mass_#*	to make a point mass node the first node of the beams.
rt1 *x y z const ;*	to create the first node of the beams in Cartesian coordinates.
cy1 ρ θ *z const ;*	to create the first node of the beams in cylindrical coordinates.
sp1 ρ θ ϕ *const ;*	to create the first node of the beams in spherical coordinates.

(Selection of the second node)

n2 *node_#*	to make an existing node the last node of the beams.
pm2 *point_mass_#*	to make a point mass the last node of the beams.
rt2 *x y z const ;*	to create the last node of the beams in Cartesian coordinates.

cy2 ρ θ *z const ;* to create the last node of the beams in cylindrical coordinates.

sp2 ρ θ ϕ *const ;* to create the last node of the beams in spherical coordinates.

(Selection of the orientation)

n3 *node_#* to make an existing node the last node of the beams.

pm3 *point_mass_#* to make a point mass the last node of the beams.

rt3 *x y z const ;* to create the last node of the beams in Cartesian coordinates.

cy3 ρ θ *z const ;* to create the last node of the beams in cylindrical coordinates.

sp3 ρ θ ϕ *const ;* to create the last node of the beams in spherical coordinates.

orient *x y z* to specify a coordinate triple to orient the beams.

sd *surface_#* to orient beam axes in the orientation of the normal of the surface

v *x y z* to orient beam axes in the direction of the vector

(Misc. options)

mate *material_#* to specify the material number.

cs *cross_section_#* to specify the cross section number (see **bsd**).

nbms *number_of_beams* to specify the number of beams in the string (default is 1).

indc *const ;* to specify the constraints on the intermediate nodes.

cur *3d_curve_#* to interpolate the string of beams along a 3D curve.

(Selection of the nodal spacing)

res *geometricratio* for relative spacing of nodes (default is equal spacing).

drs *first_geometricratio second_geometricratio* for double relative spacing of nodes.

nds *nodal_distribution_function_#* for nodal distribution by a function.

as 0 *first_thickness* first element thickness

as 1 *last_thickness* last element thickness

das *first_element_thickness last_element_thickness* first and last element thickness

sthi *sthi* for thickness in the y-direction.

sthi1 *sthi1* for thickness in the y-direction at the first end point.

sthi2 *sthi2* for thickness in the y-direction at the last end point.

tthi *tthi* for thickness in the z-direction.

tthi1 *tthi1* for thickness in the z-direction at the first end point.

tthi2 *tthi2* for thickness in the z-direction at the last end point.

csarea *csarea*	for the cross section area
sharea *sharea*	shear area
inertia *Iss Itt Irr*	inertia moments
vold *volume*	volume of Discrete Beam
lump *inertia*	lumped inertia
cablcid *system_#*	local coordinate system id number defined by the **lsys**
cabarea *area*	cable area
caboff *offset*	cable offset

(Selection of the nodal offsets)

noint	for no interior node offset interpolation
roff1 *roff1*	for x-component of offset vector for first end point.
soff1 *soff1*	for y-component of offset vector for first end point.
toff1 *toff1*	for z-component of offset vector for first end point.
roff2 *roff2*	for x-component of offset vector for last end point.
soff2 *soff2*	for y-component of offset vector for last end point.
toff2 *toff2*	for z-component of offset vector for last end point.

(Selection of the pin flags)

ldr1 *ldr1*	to release the x-translation constraint at first end point.
lds1 *lds1*	to release the y-translation constraint at first end point.
ldt1 *ldt1*	to release the z-translation constraint at first end point.
lrr1 *lrr1*	to release the rotation constraint about the x-axis at first end point.
lrs1 *lrs1*	to release the rotation constraint about the y-axis at first end point.
lrt1 *lrt1*	to release the rotation constraint about the z-axis at first end point.
ldr2 *ldr2*	to release the x-translation constraint at first end point.
lds2 *lds2*	to release the y-translation constraint at first end point.
ldt2 *ldt2*	to release the z-translation constraint at first end point.
lrr2 *lrr2*	to release the rotation constraint about the x-axis at first end point.
lrs2 *lrs2*	to release the rotation constraint about the y-axis at first end point.
lrt2 *lrt2*	to release the rotation constraint about the z-axis at first end point.
ldr3 *ldr3*	to release the x-translation constraint at first end point.
lds3 *lds3*	to release the y-translation constraint at first end point.

ldt3 *ldt3* to release the z-translation constraint at first end point.
lrr3 *lrr3* to release the rotation constraint about the x-axis at first end point.
lrs3 *lrs3* to release the rotation constraint about the y-axis at first end point.
lrt3 *lrt3* to release the rotation constraint about the z-axis at first end point.
ldp *displacement* for the initial longitudinal displacement.
theta *angle* for the orientation angle for the cross section.
warpage *first_warpage_node second_warpage_node* for two nodes used to determine warpage in the beam.

where *const* can be any of

dx	to constrain the x-displacement
dy	to constrain the y-displacement
dz	to constrain the z-displacement
rx	to constrain the x-axis rotation
ry	to constrain the y-axis rotation
rz	to constrain the z-axis rotation

Remarks

There are many options to this command. However, many of the options are specific to a single simulation code. There is some overlap, but there is little consistency among the simulation codes on beam element properties. Care must be taken in selecting the options by knowing the options needed for the target simulation code. The dialogue box makes these selections easier.

This command is functional in the Merge Phase, and it is designed to create a general collection of beams or a single beam. We recommend that you use the dialogue box for **bm**.

You can use an existing node of the mesh for a beam, specify coordinates to create a new node for a beam, or you can use a point mass as a node for a beam. Coordinates can be specified in Cartesian, cylindrical, or spherical coordinates.

Beam orientation can be defined using a third node, using a point mass, or by creating another node in Cartesian, cylindrical, or spherical coordinates. Use the output-code specific options in the MATERIAL Menu of the Control Phase to define materials for the beams.

Use the **bsd** to define a beam cross-section type.

Nodes are automatically created if the number of beams specified is greater than 1.

You can define beam elements that follow a 3D curve, and specify the number of such elements, along with a spacing rule for the intermediate nodes.

Optional thickness parameters may be specified for the first and last beams when creating multiple beams. Intermediate beams will have thicknesses that are interpolated from the end beams. You may specify offsets for the first and last nodes, and optionally interpolate these offsets to intermediate nodes.

Constraints which couple the beams to the existing mesh can be eliminated. This may be done separately for the first, last, and intermediate nodes. An initial longitudinal displacement can be specified. An optional orientation angle can be specified. Warpage nodes can be defined for codes which support such options. **Bend** geometry options can be specified for codes which support such options.

bsd global beam cross section definition

bsd *cross_section_#* **cstype** *type t_options* ;;
where *type* and *t_options* can be:

7 *t_options ;* for PIPE
where *t_options* can be
abcs1 *radius* radius of the pipe
abcs2 *thickness* wall thickness
nabip1 *#_integrations*
trss *stiffness*
abtemp *Temperature*

8 *t_options ;* for BOX
where *t_options* can be
abcs1 *width*
abcs2 *height*
abcs3 *thickness*
abcs4 *thickness*
abcs5 *thickness*
abcs6 *thickness*
nabip1 *#_integrations*
nabip2 *#_integrations*
trss *stiffness*
abtemp *Temperature*

9 *t_options ;* for CIRCLE
where *t_options* can be
abcs1 *radius*
nabip1 *#_integrations*

nabip2 *#_integrations*
trss *stiffness*
abtemp *temperature*

10 *t_options ;* for I-BEAM

where *t_options* can be

abcs1 *depth*
abcs2 *height*
abcs3 *width*
abcs4 *width*
abcs5 *thickness*
abcs6 *thickness*
Abcs7 *thickness*
nabip1 *#_integrations*
nabip3 *#_integrations*
trss *stiffness*
abtemp *temperature*

11 *t_options ;* for RECTANGLE

where *t_options* can be

abcs1 *width*
abcs2 *height*
nabip1 *#_integrations*
nabip2 *#_integrations*
trss *stiffness*
abtemp *temperature*

12 *t_options ;* for HEXAGON

where *t_options* can be

abcs1 *thickness*
abcs2 *thickness*
nabip1 *#_integrations*
nabip2 *#_integrations*
trss *stiffness*
abtemp *temperature*

13 *t_options ;* for ELBOW

where *t_options* can be

abcs1 *radius*
abcs2 *thickness*
abcs3 *radius*
nabip1 *#_integrations*
nabip2 *#_integrations*
nabip3 *#_integrations*
trss *stiffness*

abtemp *temperature*

14 *t_options ;* for TRAPEZOID

where *t_options* can be

abcs1 *width*
abcs2 *height*
abcs3 *width*
abcs4 *depth*
nabip1 *#_integrations*
nabip2 #_integrations
rss *stiffness*
abtemp *temperature*

15 *t_options ;* for I-SECTION

where *t_options* can be

abcs1 *width*
abcs2 *height*
abcs3 *thickness*
abcs4 *thickness*
nabip1 *#_integrations*
nabip2 *#_integrations*
trss *stiffness*
abtemp *temperature*

16 *t_options ;* for ARBITRARY

where *t_options* can be

cscrv *y1 z1 ... yn zn* ;
cssth thick1 ... *thickn* ;
trss *stiffness*
abtemp *temperature*

Remarks

Choose any positive integer for the identification number (*cross_section_#*). This number is used to reference the cross section definition within the **bm**, **ibm**, **ibmi**, **jbm**, **jbmi**, **kbm**, and **kbmi** commands.

ibm generate beams in the i-direction

ibm *region #_in_j #_in_k material orientation cross_section option*

where

#_in_j	is the number of columns of beam elements in the j-direction
#_in_k	is the number of columns of beam elements in the k-direction
material	is the material number
orientation	is the option of orientation of the cross section axis

j	second axis orientation in the j-direction
k	second axis orientation in the k-direction
sd *surface_#*	second axis orientation in the normal to the surface
v *xn yn zn*	second axis orientation by the vector
none	

cross_section is the cross-section definition number assigned with **bsd**

option can be

reverse	the order of the nodes is the reverse of the default
si *sid_#*	Sliding Interface Number
vold *volume*	volume of Discrete Beam
lump *inertia*	lumped inertia
cablcid *system_#*	local coordinate system id number defined by the **lsys**
cabarea *area*	cable area
caboff *offset*	cable offset
csarea *area*	cross section area
sharea *area*	shear area of cross section
inertia *iss itt irr*	cross section moments of inertia
thickness	thickness (Hughes-Liu)
roff1 x	x-component of offset vector for first end point.
soff1 y	y-component of offset vector for first end point.
toff1 z	z-component of offset vector for first end point.
roff2 x	x-component of offset vector for last end point.
soff2 y	y-component of offset vector for last end point.
toff2 z	z-component of offset vector for last end point.
ldr1	release the x-translation constraint at first end point.
lds1	release the y-translation constraint at first end point.
ldt1	release the z-translation constraint at first end point.
lrr1	release the rotation constraint about the x-axis at first end point.
lrs1	release the rotation constraint about the y-axis at first end point.
lrt1	release the rotation constraint about the z-axis at first end point.
ldr2	release the x-translation constraint at last end point.
lds2	release the y-translation constraint at last end point.
ldt2	release the z-translation constraint at last end point.
lrr2	release the rotation constraint about the x-axis at last end point.
lrs2	release the rotation constraint about the y-axis at last end point.

lrt2	release the rotation constraint about the z-axis at last end point.
ldr3	release the x-translation constraint at intermediate point.
lds3	release the y-translation constraint at intermediate point.
ldt3	release the z-translation constraint at intermediate point.
lrr3	release the rotation constraint about the x-axis at intermediate points.
lrs3	release the rotation constraint about the y-axis at intermediate points.
lrt3	release the rotation constraint about the z-axis at intermediate points.
theta θ	orientation angle for the cross section.
warpage *n1 n2*	two nodes used to determine warpage in the beam.
geom *option*	method of determining curvature

where *option* can be

1 for center of curvature
2 for tangent of centroid arc
3 for bend radius
4 for arc angle

Remarks

This command is available only in the **block** or **cylinder** Part Phase. This command generates an array of beam elements conforming to the geometry and nodes of a solid or shell regions in the i-direction.

This feature is useful in generating structural elements embedded within the solid or shell region.

The local coordinate orientation can be selected in many ways or none at all.

The *v* option specifies a vector for the orientation. That vector is defined by the coordinate system. If the part is a cylinder, the vector is in the form of a radial, angular, and z-offset. Depending on the coordinates of the beam, the cylindrical vector will define a different orientation for each beam since the vector offset is made in cylindrical coordinates and then transformed to Cartesian coordinates. Each beam element can have an additional third node used to determine the orientation of the cross-section and local material coordinate system. The neighboring beam elements can be used to select the orientation node. The options **i**, **j**, or **k** will select the node of the corresponding neighboring beam element. In each case, only two of the options are appropriate.

The **sd** option is used to orient the beam normal to a surface. The **v** option creates an orientation in a given vector direction. In the latter two cases, a new node is created for each beam, when nodes are required to orient beams. Use the **orpt** command when using the **sd** option.

To define the cross-section, use the **bsd** command.

A 1D sliding interface can be specified for each string of beams. Only the first sliding interface is specified. The remainder are assumed to follow in sequence. Use **sid** command to define each sliding interface.

```
sid 1 rebar;;sid 2 rebar;;sid 3 rebar;;sid 4 rebar;;
block 1 3 5;1 3 5;1 3 5;1 3 5;1 3 5;1 3 5;
ibm 1 1 1 3 3 3 2 2 1 j si 1 1 ;
```

In the above example, 4 rebar sliding interfaces are generated between 4 strings of beam elements and the corresponding brick elements, respectively. Since this is a sliding interface, there are new nodes automatically generated for the beam elements so that the beams are not coupled to the solid elements except through the sliding interface. Care should be taken not to merge these additional nodes out in the merge phase. They automatically will not be merged with their equivalent solid element nodes with the same coordinates, but they can be merged to other parts of the mesh. Use dummy sliding interfaces to control the merging.

Many of the options are designed for a specific simulation code or for a specific beam type. There is some overlap in that some of the options are used for several different types or simulation codes. Because of this complexity, you are advised to use the dialogue box to make your selection of options when using this command. The options override the properties given by the **bsd**. See also **bm**, **bsd**, and **orpt** commands.

ibmi generate beams in the i-direction by index progression

ibmi *progression #_in_j #_in_k material orientation cross_section option*

Remarks

See ibm for the details and remarks.

jbm generate beams in the j-direction

jbm *region #_in_i #_in_k material orientation cross_section option*

Remarks

See ibm for the details and remarks.

jbmi generate beams in the j-direction by index progression

jbmi *progression #_in_i #_in_k material orientation cross_section option*

Remarks

See ibm for the details and remarks.

kbm generate beams in the k-direction

kbm *region #_in_i #_in_j material orientation cross_section option*

Remarks

See ibm for the details and remarks.

kbmi generate beams in the k-direction by index progression

kbmi *progression #_in_i #_in_j material orientation cross_section option*

Remarks

See ibm for the details and remarks.

npm creates a node with a point mass (part phase)

npm *mp_node_# x y z mass options ;*

where

mp_node_# is the node number which is created,
x y z are the coordinates of the point mass,
mass is the assigned mass, and
options can be :

inc *increment* for the increment in the node number under replication,
dx for no nodal displacement in the x-direction,
dy for no nodal displacement in the y-direction,
dz for no nodal displacement in the z-direction,
rx for no nodal rotations about the x-axis,
ry for no nodal rotations about the y-axis,
rz for no nodal rotations about the z-axis,

mdx	for no mass displacement in the x-direction,
mdy	for no mass displacement in the y-direction,
mdz	for no mass displacement in the z-direction,
mrx	for no mass rotation about the x-axis,
mry	for no mass rotation about the y-axis,
mrz	for no mass rotation about the z-axis,
ixx *mom*	to specify the moment of inertia about the x-axis,
iyy *mom*	to specify the moment of inertia about the y-axis,
izz *mom*	to specify the moment of inertia about the z-axis,
pdamp *alpha*	for the proportional damping factor (ABAQUS), and/or
cdamp *fraction*	for the fraction of critical damping (ABAQUS).

Remarks

This new node can be attached to the mesh by creating a spring using the **spring** command in the Part or Merge Phase, or by creating a beam in the Merge Phase using the **bm** command. This new node can also be attached to the rest of the mesh in the Merge Phase by merging it to a neighboring node (see **t**, **tp**, **stp**, **bptol**, and **ptol**). This is distinguished from the assignment of a mass to a vertex of the present part. The latter can be done using the **pm** command. In both cases, the point mass is replicated or transformed along with the present part (see **lrep**, **grep**, and **pslv**). In order to create a new node and assign it a point mass such that it does not get replicated or transformed along with the present part, then use the **npm** command in the Merge Phase. In order to assign a point mass to any node in the mesh such that it does not get replicated or transformed along with the present part, use the **pm** command in the Merge Phase.

npm creates a new node and assigns a point mass to it (merge phase)

npm *mp_node_# x y z mass options ;*

where

an *option* can be:

dx	no nodal displacement in the x-direction
dy	no nodal displacement in the y-direction
dz	no nodal displacement in the z-direction
rx	no nodal rotations about the x-axis
ry	no nodal rotations about the y-axis
rz	no nodal rotations about the z-axis
mdx	no mass displacement in the x-direction
mdy	no mass displacement in the y-direction
mdz	no mass displacement in the z-direction
mrx	no mass rotations about the x-axis

mry	no mass rotations about the y-axis
mrz	no mass rotations about the z-axis
ixx *mom*	specify the moment of inertia about the x-axis
iyy *mom*	specify the moment of inertia about the y-axis
izz *mom*	specify the moment of inertia about the z-axis
pdamp *alpha*	proportional damping factor (ABAQUS)
cdamp *fraction*	fraction of critical damping (ABAQUS)

Remarks

This newly created node is separate from the existing mesh and can be attached by generating a beam or spring using this new node (see **bm** or **spring**). It can also be attached to the rest of the mesh by merging it to a neighboring node (see **t**, **tp**, **stp**, **bptol**, and **ptol)**. This is distinguished from assigning a mass to an existing node of the mesh. The latter can be done using the **pm** command. To create a new node and assign it a point mass such that it is replicated or transformed along with the part, then use the **npm** command in the Part Phase (see **lrep**, **grep**, and **pslv**). To assign a point mass to a vertex of a part such that it is replicated or transformed along with the part, use the **pm** command in the Part Phase. All of the options are not needed by all output options.

offset add offset to numbered entities in the output

offset *type offset*

where *type* can be

nodes	node numbers
bricks	brick elements (or all elements)
nsetoff	node sets if they are automatically numbered (not named)
fsetoff	face sets if they are automatically numbered (not named)
esetoff	element sets if they are automatically numbered (not named)
partoff	parts

Remarks

The **nodes**, **bricks**, **nsetoff**, **esetoff**, **partoff,** and **nsetoff** options affect the automatically numbered node sets as a result of the **fc**, **fd**, **fv**, **ft**, **acc**, and **mom** nodal boundary conditions. **Esetoff** affects the automatically numbered element sets as a result of the **pr** condition. **Partoff** affects the automatic numbering of element sets based on the part number.

plane define a boundary plane

plane *plane_#* x_0 y_0 z_0 x_n y_n z_n *tolerance* **symm**

Remarks

This command is used to define nodal constraints for nodes on a symmetry plane. The point (x_0, y_0, z_0) is on the symmetry plane with a normal vector (x_n, y_n, z_n). Nodes are automatically selected for the symmetry plane constraint if they are within the specified tolerance of the plane.

The symmetry feature is complicated, depending on the type of plane and the simulation code. If the symmetry plane is parallel to one of the planes where x=0, y=0, or z=0, then the nodes on the symmetry plane are assigned constraints in the global coordinate system. These types of symmetry planes are referred to as canonical symmetry planes and are equivalent to the following constraints:

plane parallel to x=0: x-displacement, y-rotation, z-rotation
plane parallel to y=0: y-displacement, x-rotation, z-rotation
plane parallel to z=0: z-displacement, x-rotation, y-rotation

Nodes on non-canonical symmetry planes are constrained in local coordinate systems.

pm point mass to a vertex of the present part (part phase)

pn *region node_mass options ;*

where
node_mass is the assigned mass, and
options can be :

mdx	for no mass displacement in the x-direction,
mdy	for no mass displacement in the y-direction,
mdz	for no mass displacement in the z-direction,
mrx	for no mass rotations about the x-axis,
mry	for no mass rotations about the y-axis,
mrz	for no mass rotations about the z-axis,
ixx *mom*	to specify the moment of inertia about the x-axis,
iyy *mom*	to specify the moment of inertia about the y-axis,
izz *mom*	to specify the moment of inertia about the z-axis,
pdamp *alpha*	for the proportional damping factor (ABAQUS), and/or
cdamp *fraction*	for the fraction of critical damping (ABAQUS).

Remarks

This is distinguished from a node which is created separate from the mesh, assigned a mass, and then later attached to the mesh by a beam or spring. This latter type of point mass is created using the **npm** command, above. The **pm** point mass is replicated along with the present part (see **lrep**, **grep**,

and **pslv**). In order to assign a point mass to any node in the mesh such that it does not get replicated or transformed along with the present part, use the **pm** command in the Merge Phase. In order to create a new node and assign it a point mass such that it does not get replicated or transformed along with the present part, then use the **npm** command in the Merge Phase.

pm assigns a point mass to a node of the mesh (merge phase)

pm *node_# mass otions ;*

where

an *option* can be:

mdx	no mass displacement in the x-direction
mdy	no mass displacement in the y-direction
mdz	no mass displacement in the z-direction
mrx	no mass rotations about the x-axis
mry	no mass rotations about the y-axis
mrz	no mass rotations about the z-axis
ixx *mom*	specify the moment of inertia about the x-axis
iyy *mom*	specify the moment of inertia about the y-axis
izz *mom*	specify the moment of inertia about the z-axis
pdamp *alpha*	proportional damping factor (ABAQUS)
cdamp *fraction*	fraction of critical damping (ABAQUS)

Remarks

This is distinguished from creating a new node separate from the mesh and assigning a mass to it. The latter can be done using the **npm** command. To assign a point mass to a vertex within a part such that it is replicated or transformed along with the part, use the **pm** command in the Part Phase (see **lrep**, **grep**, and **pslv**). In order to create a new node and assign it a point mass such that it is replicated or transformed along with a part, then use the **npm** command in the Part Phase. All of the options are not needed by all output options.

sid sliding interface definition

sid *slide_# option* ;

where the *option* can be

dummy	nodes in this interface will not be merged
sv	sliding with voids
dni	discrete nodes impacting surface
inter *params*	interface elements

where a *param* can be

fric *friction_factor*	static coefficient of friction
fric2 *friction_factor*	anisotropic friction coefficient
stif *stiffness*	stiffness in stick
essl *stress*	equivalent shear stress limit

Remarks

Sliding interfaces or contact surfaces are constructed in 3 steps. These steps can be done in any order.

1. define the properties
2. select the slave side
3. select the master side

The **sid** command is used to define the properties. The **si** and **sii** commands are used in the part phase or the merge phase to select the nodes or faces that form the master and slave sides of the interface.

Alternative to using the **sid**, **si**, and **sii** commands, one can construct a face set. This will be written to the output file as a set. Then it is a simple matter to add the keyword command to the output file using a text editor to transform that set into a contact surface or sliding interface. This approach has the problem that nodes may be merged across the two sides because they are not defined as sliding interfaces.

When nodes are merged, nodes across a sliding interface will not be merged. When a merge command is first issued in the merge phase, a table is written listing the number of nodes and faces associated with each sliding interface.

The **dummy** type interface is actually used to avoid merging of nodes. A sliding interface of this type is not written to the output file.

The nodes and faces of a sliding interface or contact surface can be viewed in the merge phase using the **si** option of the **co** command.

si assign sliding interface to region (part phase)

si *region sliding_# type*

where

sliding_# reference number for the interface

where *type* can be

m for master

s for slave

Remarks

This command, and its relative **sii**, specify that faces in the mesh are part of a sliding interface. You can use these commands to assign a shell or brick face to a sliding interface definition. In order to define the properties of the sliding interface, first use the command **sid**. **Sid** defines the properties of the sliding interface that you refer to in **si** and **sii**.

Surfaces from 3D solid brick elements have an obvious orientation pointing outward. However, this is not the case with sliding interfaces on 2D shell surfaces. You can provide information about how to orient them. That is the purpose of the **orpt** command.

During the node merging process using, using **stp** for example, **True*Grid*®** will not merge nodes on opposite sides of a sliding interface.

Use the merge phase command **co** with the **si** option to view the numbered sliding interfaces and their orientation.

si select nodes or faces for a sliding interface (assembly phase)

si fset *fac_set interface_# type ;*

where *type* can be one of

m	master side of the interface
s	slave side of the interface

Remarks

The global properties of a sliding interface are defined using the **sid** command. The dummy sliding interface type, is used to control the merging without the side effect of causing a sliding interface definition in the output.

Use the **fset** or **fseti** commands to create a face set. You can also use the interactive set selection feature in the merge phase found in the Environment window with the ***Pick*** and ***Sets*** buttons.

sii assign sliding interfaces (part phase)

sii *progression sliding_# type*

where *type* can be

m	for master
s	for slave

Remarks

See the **si** (part phase) remarks.

IV. INDEX