FabTips

Abaqus

Abaqus v. 6.7-1 and Windows Vista Business were used

Installation

This information is from 2008 and may be outdated

In order to code your own Fortran subroutines you need to install Microsoft Visual C++ (or Studio) and Intel Fortran Compiler. Check which versions are required for you Abaqus version. For Abaqus 6.7-1 you need VC++ 2003 or 2005 and Fortran 8.1 or 9.1. At NTNU we have a more recent version of Fortran. In order to install an older version, register an account for this version at:

https://registrationcenter.intel.com/ [https://registrationcenter.intel.com/]

then from the product page of your account, click the download link corresponding to your product. On the download page there is a drop-down menu from which you can choose an older version.

Check that the compilers are installed correctly:

```
% abaqus verify -install
```

The VC++ and Fortran test result should be 'Pass'. If it's not check the error message. I had to edit several environment variables by hand. The lists of variables are found in batch files in the installation folders of the product. The error message tells you their names.

Tips: it is a good idea to add the abaqus folder to the PATH so you can call

```
% abaqus
```

from any location.

Basics

Getting started

Start with the tutorial Build your own model while you're learning the tutorial. Rather sooner than later. Then look for relevant examples in the manual and build the corresponding models.

Extract .inp input files

```
% abaqus fetch job=archive.inp
```

Note: the inp files linked to from the example pages are already inflated

Import .inp input files

File > Import > Model…

Elements denomination

R- : rigid

B- : beam

ABAQUS/Standard VS. ABAQUS/Explicit

[Read also paragraph 2.4 in the Abaqus Theory Manual]

Abaqus/Standard assumes small deformations while Abaqus/Explicit assumes large deformations From Tutorial Ch.9:

“Explicit methods require a small time increment size that depends solely on the highest natural frequencies of the model and is independent of the type and duration of loading. Simulations generally take on the order of 10,000 to 1,000,000 increments, but the computational cost per increment is relatively small.

The term “explicit” refers to the fact that the state at the end of the increment is based solely on the displacements, velocities, and accelerations at the beginning of the increment.

Implicit methods do not place an inherent limitation on the time increment size; increment size is generally determined from accuracy and convergence considerations. Implicit simulations typically take orders of magnitude fewer increments than explicit simulations. However, since a global set of equations must be solved in each increment, the cost per increment of an implicit method is far greater than that of an explicit method.”

Vocabulary

pinned joint: allows the joined members to swivel. Cannot transmit rotation (torque), only push or pull

Misc

. rigid surface: a rigid body reference point must be created:

Tools > Reference Point

. change working folder so that system files are stored in the same folder as the model database: File > Set work directory

Output

Field Output Requests: for variables that should be written at relatively low frequencies

History Output Request: for variables that should be written at high frequency from a small portion of the model, e.g. the displacement of a single node

Visualisation

'Display Groups' controls what parameter you visualise

Creating and running a model

Steps:

- define parts

- define materials

- define sections

- assign section properties (Part Name > Section Assignments)

- create instances (Assembly > Instances)

- define steps

- define output

- apply boundary conditions

- create the mesh

- create a job

- checking the model

Nlgeom ON if the loads result in large displacements. If turned ON at one step, it must be ON for subsequent steps. Possible to replace steps, e.g. to run a different simulation on same objects: Step > Replace

Shortcuts

F2: Pan view

F3: Rotate view

F4: Magnify view

F5: Box zoom

F6: Auto-fit view

Advanced modelling techniques

Contact (Getting Started - Ch. 12)

. additional steps:

- Assembly > Surfaces

- Interaction properties (e.g. friction)

- Interactions

. First-order elements should be used for contact simulations

Surfaces

. master vs. slave surface - rules:

- slave surface should be the more finely meshed surface

- if similar mesh densities, slave surface should be that of the softer material

. better to use first-order elements for the slave surface

. rigid surface is always the master surface

. rigid surface should be large enough to ensure that slave nodes do not slide off and “fall behind” the surface

. the deformable mesh must be refined enough to interact with any feature on the rigid surface

. the master surface must be smooth. Analyitcal rigid surfaces can be smoothed by defining a fillet radius that is used to smooth any sharp corners in the rigid surface definition

. 2D surfaces: Abaqus/CAE distinguishes the two sides of the surface using arrowheads

. constraints should not be applied to regions that are also part of a contact surface

User-defined subroutines

= Compile =

If a user-defined (Fortran) subroutine doesn't compile, the job will abort and Abaqus returns a message that there was a problem during compilation. In order to obtain the compiler message, try and compile the subroutine from the command-line:

ifort /c /Gm /recursive /nologo /include:%I subroutine.for

= Debug =

The best way I've found so far to debug user-defined subroutines is to include a line similar to:

```
write (*,*) 'Debug variable = ', myVar
```

The output is written to the file [jobname].log

= nblock =

The nblock variable is the number of points to be processed in this call to the subroutine. Note that not all points are necessarily processed in the same call, i.e. the subroutine may be called several times for a give time step.

Buoyancy

(The following gives you directions for modelling buoyancy in Abaqus but it doesn't work satisfyingly. I'm working on the issue and will update the paragraph as soon as I've found a good solution.)

Abaqus/Explicit

step: dynamic, explicit

define the surface to which the buoyancy pressure will be applied (i.e. the outer surface of the solid)

create a user-defined pressure load and apply it to this surface

create a fortran file called vdload.for with the following code:

[code will be uploaded come another time]

. Edit the job. In the General tab, fill the path to the subroutine Fortran file

. Copy Abaqus\site\vaba_param_sp.inc (single-precision) or Abaqus\site\vaba_param_dp.inc (double-precision) to the job folder and rename it vaba_param.inc

. Submit the job

. Make sure the Fortran file compiles without errors. You may compile the file beforehand with the following command:

```
% ifort -c vdload.for
```

. A common source of error is that your PATH, LIB and INCLUDE environment variables are not set correctly.

. Don't forget to apply a gravity load (body force).

. You may also call the job from the command line

```
% abaqus job=myjob user=routine.for input=inputfile.inp
```

In this case you first need to create the input file (right-click on the job > Write input).

When doing so the .odb file is named myjob.odb. It is recommended to use the job name from the cae file.

Drag pressure

The drag pressure dissipates the energy of a body subjected to buoyancy. Without drag pressure an ice block that isn't in equilibrium at simulation start will oscillate perpetually around equilibrium.

We choose a pressure that is proportional to the square of the velocity of the bottom surface and is directed in the direction opposite to movement:

```
P = - rhoIce * dragCoeff * velocity^2
```

Tests should be made to find a suitable value for the drag coefficient, i.e. a value that dampens oscillations rapidly compared to the total duration of the simulation and at the same not so much that it prevents the ice with moving with e.g. the tide.

Plastic fracturing

. define plastic behaviour (under Materials)

. under Field Output Requests, turn on the Status variable in the State/Field/User/Time section

The value of the STATUS field output variable determines whether or not an element has failed. Failed elements are removed from the model plots.

. write the input file (right-click on the job name)

Edit the file and add the following lines under '** MATERIALS'

```
*Shear failure

1.0e-3
```

where 1.0e-3 is the equivalent plastic strain at failure (See also the keyword manual for the *Shear failure field) . There exist more complicated failure criteria. See material models; they also give which flag to use in the input file . Note: when you suppress elements some mass is removed. For that reason it is preferable to have elements that are thin in the direction perpendicular to the crack

Multiple user-defined subroutines

Apparently you need to group all subroutines into one Fortran file - not tested yet

Advanced modelling issues

Shear locking

See Getting Started 4.1

Affects the fully integrated, linear elements subjected to bending loads. Because the edges of these elements aren't able to curve, shear stresses arise.

⇒ Fully integrated, linear elements should be used only when you are fairly certain that the loads will produce minimal bending in your model.

Thin vs. thick shell elements

See Getting Started 5.2

A shell made of a single isotropic material with a ratio greater than 1/15 is considered “thick”; if the ratio is less than 1/15, the shell is considered “thin”. In laminated composite shell structures the ration should be much smaller for “thin” shell theory to apply.

Plasticity

[Tutorial, 10.2.3] When defining plasticity data in Abaqus, you must use true stress and true strain. Abaqus requires these values to interpret the data correctly. Check that section to see how to convert nominal stress/strain values to true stress/strain values.

Selecting elements for elastic-plastic problems

Plastic deformations are associated with incompressible material behaviour. For this reason not all elements can be used to model elastic-plastic problems. Incompressibility adds kinematic constrains. When elements cannot resolve all of these constraints, they suffer from volumetric locking, which causes their response to be too stiff.

Suited elements:

- fully integrated, first-order

- first-order reduced-integration solid elements

- second-order reduced-integration hybrid solid elements if strain < 20%

Results analysis

- Check that the results are almost unaffected by mesh refinement.

- Check the job diagnostics: in the visualization module select Tools > Job Diagnostics

---