Altair

HyperWorks

Altair HyperStudy 2019 Tutorials

HS-1550: Shape Optimization Study Using HyperMesh and Abaqus

HS-1550: Shape Optimization Study Using HyperMesh and Abaqus

This tutorial demonstrates how to perform a shape optimization using HyperMesh and HyperStudy. You will be using the finite element solver Abaqus, and HyperMorph to do the shape parameterization. This tutorial also demonstrates how to solve a problem when HyperMesh and HyperStudy are running in Windows and the solver is on a UNIX platform.

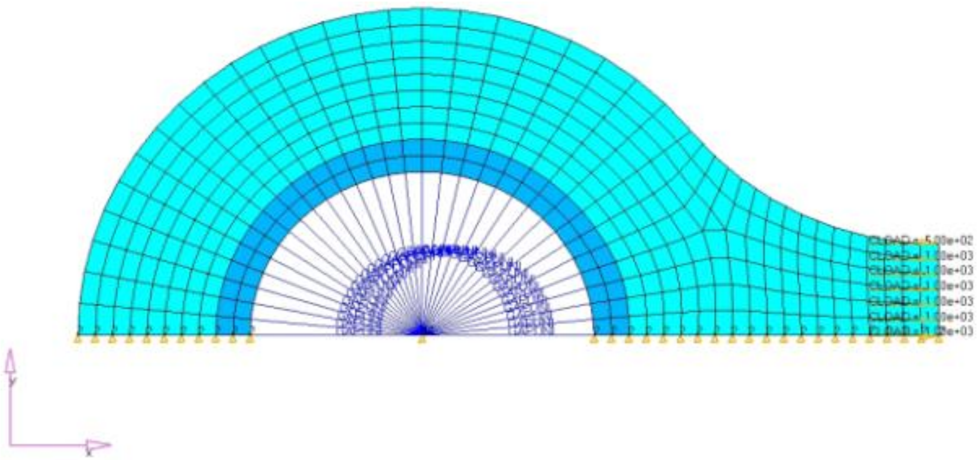
The files used in this tutorial can be found in `<hst.zip>/HS-1550/`. Copy the tutorial files from this directory to your working directory. The tutorial directory includes the following files:

`link.hm`	Original HyperMesh file.
`HS-1550_solverScript.py`	Basic Solver Script Setup (ONLY for LOCAL Machine) – Run everything on local machine (Windows or Unix)
`run_abaqus.sh`	(ONLY for REMOTE Machine) - Shell script which is designed to run the abaqus solver in the UNIX machine.
`ssh_remote.bat`	Advanced Solver Script Setup (ONLY for REMOTE Machine) – Run HyperStudy on Windows with Study Directory in Unix and the Solver is Running on Unix. Sample execution script to run Abaqus on UNIX.

In this tutorial, you will:

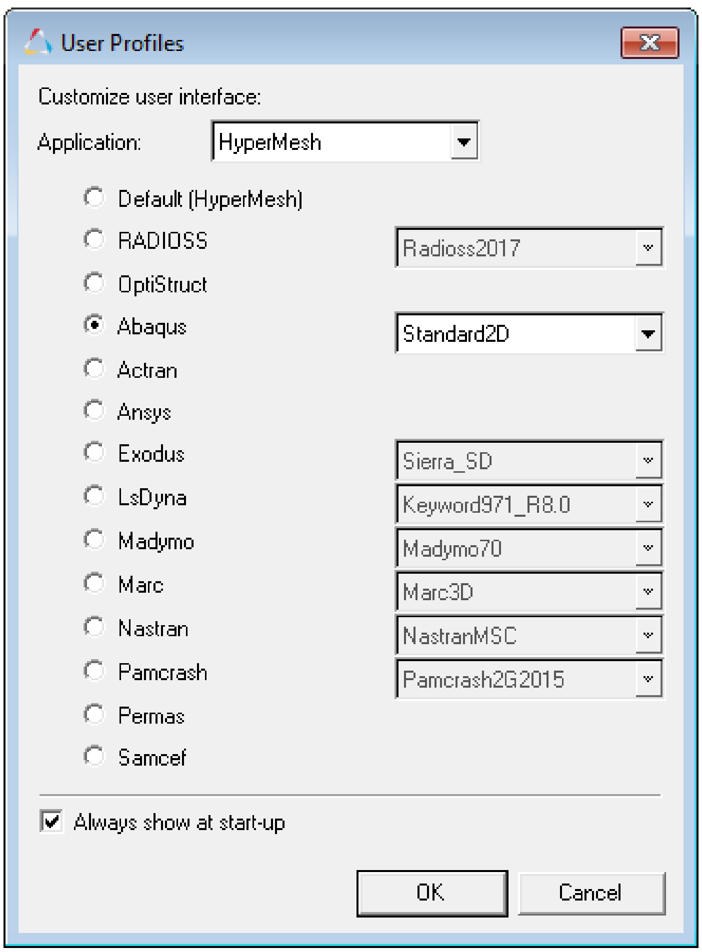
- Use HyperMorph to generate a shape variable.
- Run a study from inside HyperMesh.
- Perform a shape parameterization using HyperStudy.
- Set up a study.
- Write a script to run Abaqus on UNIX and register the script in the preference file.
- Run an optimization study.

The objective of this tutorial is to minimize the mass of a link that is connected to a shaft, given a stress constraint of 200MPa. The input variables are defined by the outer shape.



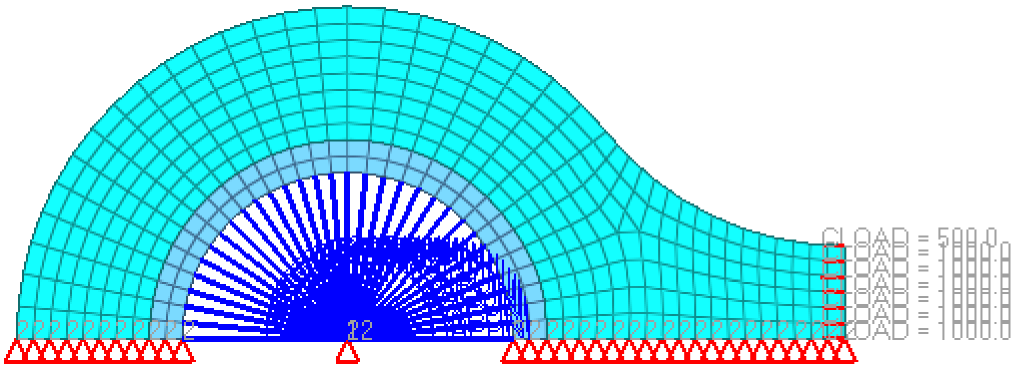
Step 1: Load the Model in HyperMesh

1. Start HyperMesh Desktop.
2. In the **User Profiles** dialog, set the user profile to ***Abaqus***, ***Standard2D***.



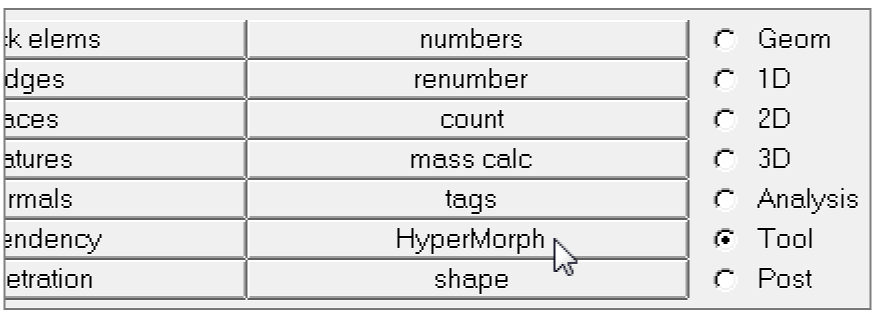
3. From the menu bar, click ***File*** > ***Open*** > ***Model***.

4. In the **Open Model** dialog, open the `link.hm` file. A finite element model appears in the graphics area.

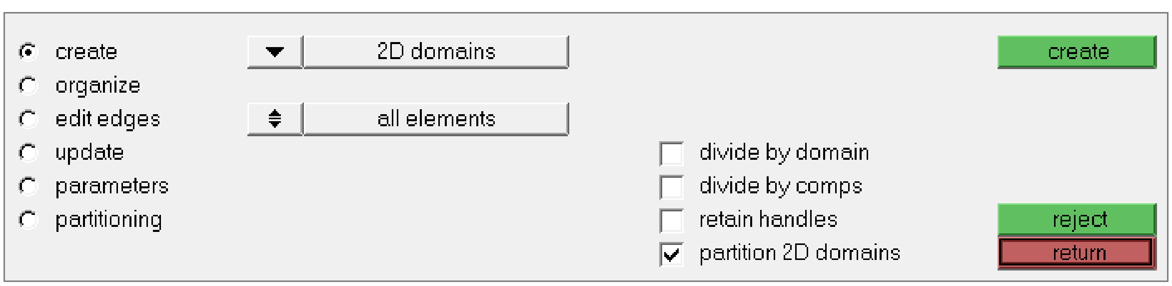


Step 2: Do the Shape Parameterization in HyperMorph

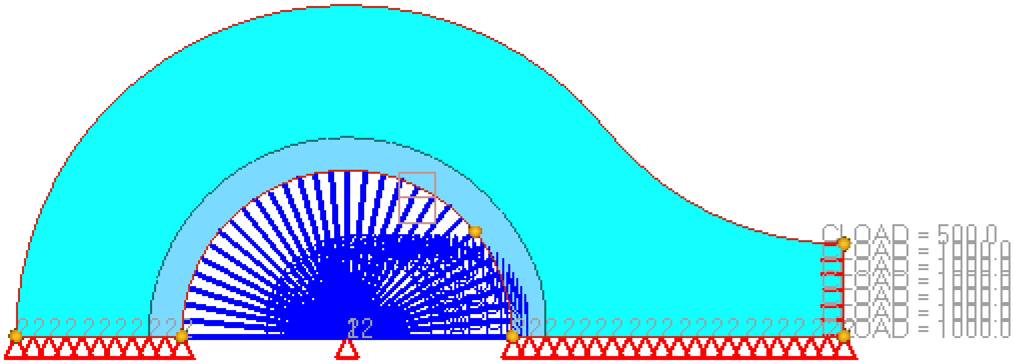
1. From the **Tool** page, click ***HyperMorph***.



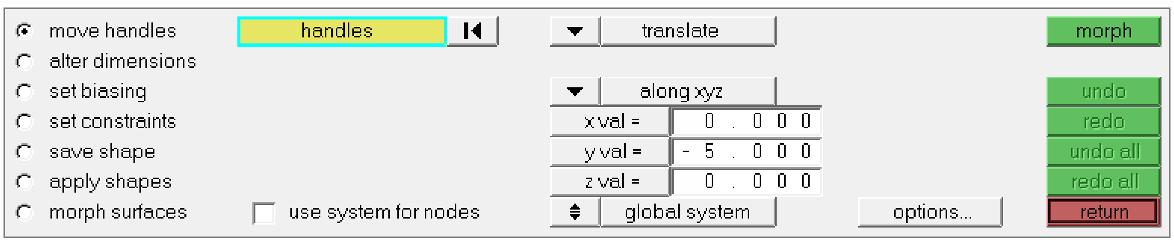
2. Click ***domains***. The **Domains** subpanel opens, from which you can create domains for the shape parameterization.
3. Click the first arrow and select ***2D domains***.
4. Click the toggle and select ***all elements***.



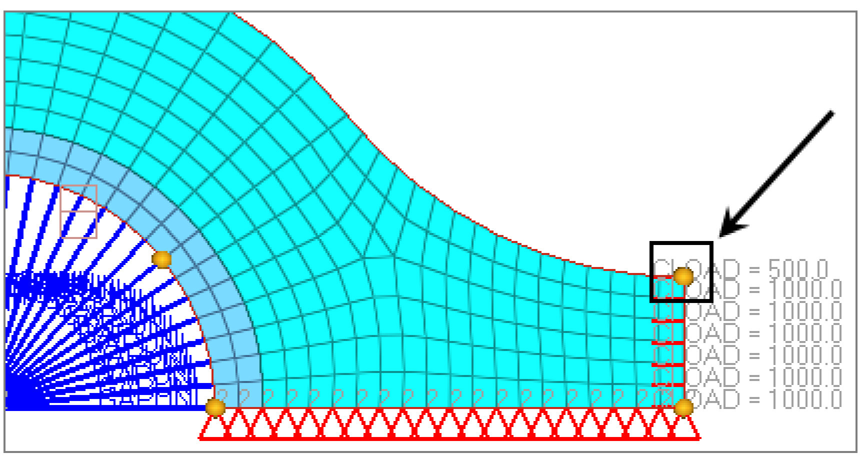
5. Click ***create***. Domains and handles are generated and will be used to manipulate the shape of the mesh and to generate shape perturbations required for the shape optimization.



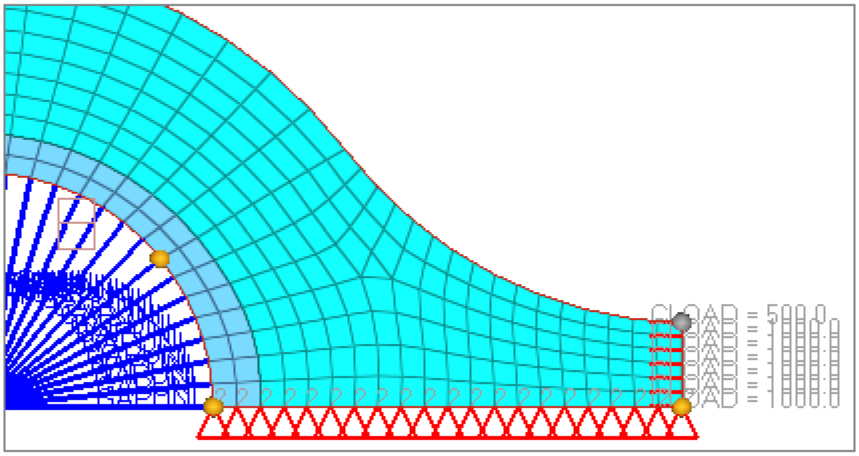
6. Click ***return***.
7. From the **HyperMorph** panel, click ***morph***. The **Morph** subpanel opens, from which you can morph the shape of the mesh. The goal of the following steps is to create one input variable for the outer edge of the link.
8. Click the second switch and select ***translate***.
9. In the **y val=** field, enter `-5.0`.



10. In the graphics area, click the yellow handle located at the top-right corner of the link as indicated in the image below.



11. Click ***morph***. The first shape is generated.



12. Click ***save shape***.
13. In the **name=** field enter, `sh1`.
14. Click ***save***.

Step 3: Basic Solver Script Setup (ONLY for LOCAL Machines) - Run everything on local machines (Windows or Unix)

To run the solver on a local machine, use the supplied python script `HS-1550_solverScript.py`, which calls the Abaqus solver using arguments specified that you specify.

Change the path in the file to the Abaqus executable on your machine.

Any additional arguments, such as memory requests, can be modified in the script as well.

```
# import statements
import os
import platform
import subprocess
import sys

# user edits---------------------------------------------------------
----------

# set abaqus path
abaqus_exe = '/my/path/to/executable/abaqus.exe'

# set abaqus arguments
abaqus_arguments = 'job=' + sys.argv[1] + ' ' + 'memory=200Mb
interactive'
```

```
# set log file name
logFile = 'logFile.txt'

# -------------------------------------------------------------------
----------

# open log file
f = open('logFile.txt', 'w')
f.write('Hello world')

# get environment information
plat = platform.system()
hst_altair_home = ''
strEnvVal = os.getenv('HST_ALTAIR_HOME')
if strEnvVal:
    hst_altair_home = strEnvVal
else:
    f.write('%EXA_xx-e-env, Environment variable not defined (
HST_ALTAIR_HOME )')

abaqus_exe = os.path.normpath(abaqus_exe)
lstCommands = [abaqus_exe, abaqus_arguments]

# echo log information
f.write('platform: ' + plat + '\n')
f.write('hst_altair_home: ' + hst_altair_home + '\n')
f.write('abaqus command: ' + abaqus_exe + '\n')


#execute the command
f.write('Running the command:\n' + ' '.join(lstCommands) + '\n')
p1 = subprocess.Popen(lstCommands, stdout=subprocess.PIPE ,
stderr=subprocess.PIPE)

# log the standard out and error
f.write('\nStandard out' + '\n' + p1.communicate()[0] + '\n')
f.write('\nStandard error' + '\n' + p1.communicate()[1] + '\n')

# close log file
f.close()
```

(OPTIONAL - ONLY for REMOTE Machine): Advanced Solver Script Setup – Run HyperStudy on Windows with Study Directory in Unix and the Solver is Running on Unix.

In order to run commands on a remote Unix machine, a local program must be installed to communicate the commands remotely. In this example, the program ssh is being used, but other equivalent or better programs exist. In most cases, the program's protocols require authentication from a password. For this setup to work the environment needs to be configured to work without an active password entry. This setup may require help from your network administrators.

The `ssh_remote.bat` file is a sample (Windows) batch file to run a script on Unix (`run_abaqus.sh`) from a HyperStudy session running on Windows.

```
ssh user_name@unix_machine  "./run_abaqus.sh %1 %HST_APPROACH_VARNAM
% %HST_STEP_INDEX% %HST_APPROACH_MODELS%"
```

The batch file uses the ssh command to log onto the Unix machine and execute the solver on the files created by HyperStudy. This script will be registered in HyperStudy as the solver script.

To use the sample script, change the following to match your setup:

1. Change the generic parameter "`unix_machine`" to match the name of the remote Unix machine on your network.
2. Change the generic parameter "`user_name`" to match your logon on the remote machine.

Note: As mentioned above, this logon should have been configured to work without a password between these two machines.

The file `run_abaqus.sh` is a shell script which is designed to run the Abaqus solver on a UNIX machine. This file should be placed in your HOME directory on the Unix machine.

```
#!/bin/sh

#constuct the model directory
approachName=$1
runNum=`printf %04i $2`
modelString=$3
array=(${modelString//:/ })
modelName=${array[0]}
myDir=~/${approachName}/run_${runNum}/${modelName}

#change the directory to model directory
cd $myDir

#change the format from windows to unix
dos2unix $1 $1
```

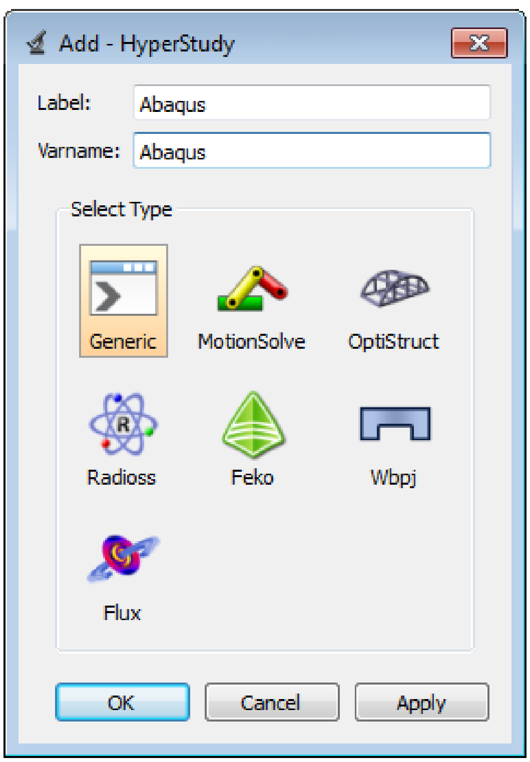
```
#edit this line to match your solver path and the appropriate
arguments
/my/path/to/solver/example/abaq631 job=$1 memory=200Mb interactive
```

Once there, perform the following:

1. Change the text formatting to be Unix compatible using the command `dos2unix` on the file: `dos2unix run_abaqus.sh`.
2. Make sure the file has executable permission. Type `chmod 755 run_abaqus.sh`.
3. Edit the `run_abaqus.sh` file and modify the path to the executable and any other options to this command as needed.

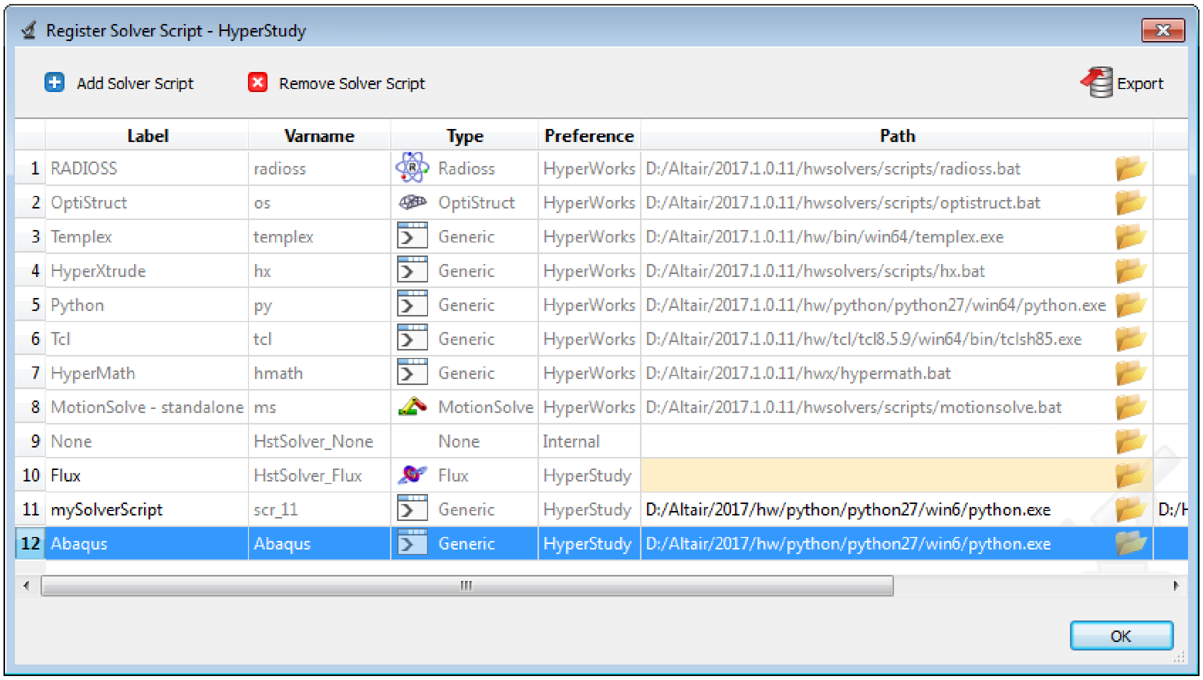
Step 4: Register Abaqus as a Solver

1. Start Hyperstudy.
2. From the menu bar, click ***Edit*** > ***Register Solver Script***.
3. In the **Register Solver Script** dialog, click ***Add Solver Script***.
4. In the **Add - HyperStudy** dialog, enter `Abaqus` in the **Label** and **Varname** fields.
5. For solver script type, select ***Generic***.
6. Click ***OK***.



7. In the **Path** column of the script **Abaqus**, click .
8. In the **Open** dialog, open the `python.exe` file.

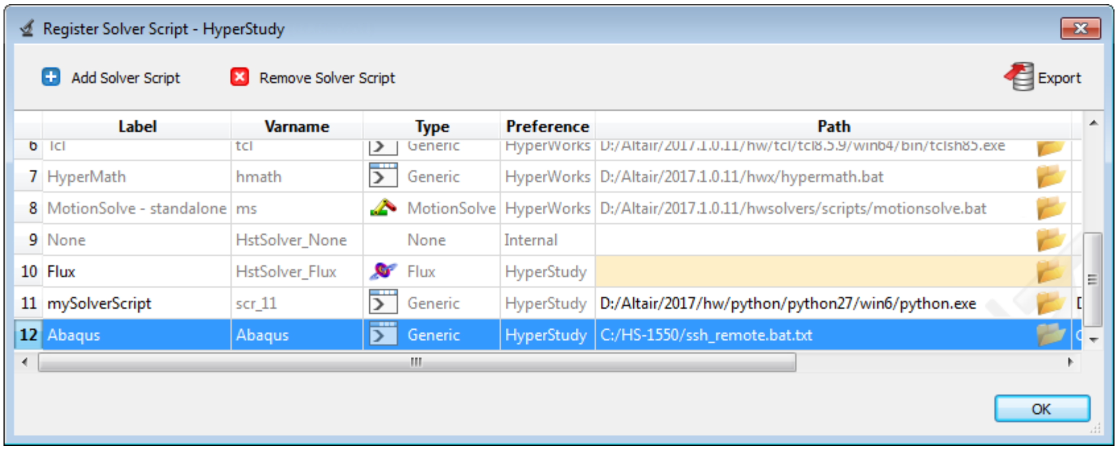
   Note: You can also copy and paste the same path and file from the Python **Path** field.



9. In the **Argument** column of the Abaqus script, click .
10. In the **Open** dialog, navigate to your working directory and open the `HS-1550_solverScript.py` file.
11. In the top, right corner of the **Register Solver Script** dialog, click ***Export***.
12. In the **Save Preferences** dialog, navigate to your working directory.
13. In the **File name** field, enter a label for the new user preference file (example: `userprefs.mvw`).
14. Click ***Save***.

    Note: Do not overwrite the system preferences file, which is located in `<install_directory>/hw` by default.

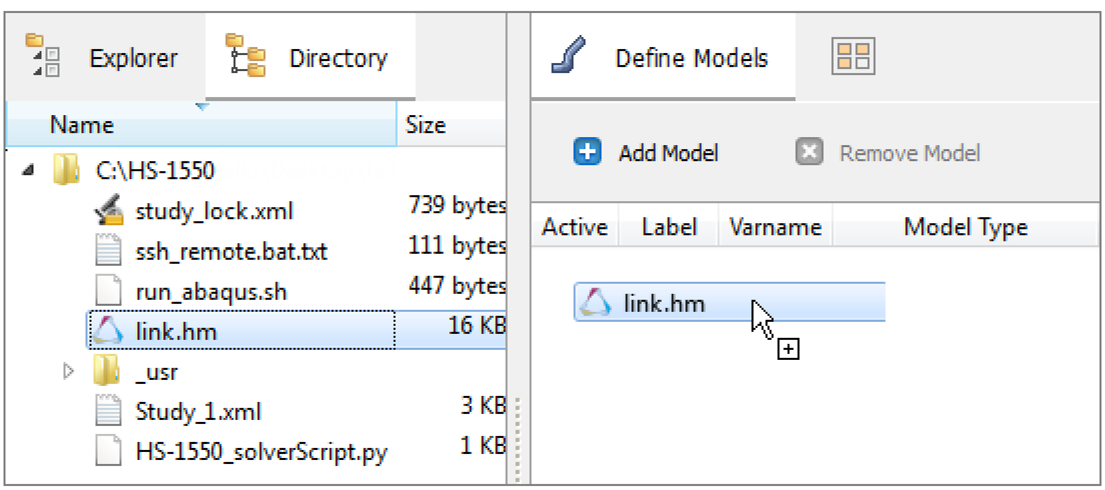
15. **Optional**. ONLY for REMOTE Machines: Register Abaqus as a solver.



Step 5: Perform the Study Setup

1. To start a new study, click ***File*** > ***New*** from the menu bar, or click on the toolbar.
2. In the **HyperStudy – Add** dialog, enter a study name, select a location for the study, and click ***OK***.

   Note: The study directory MUST be your home on the mapped UNIX machine.
3. Go to the **Define Models** step.
4. Add a HyperMesh model.
   a. From the **Directory**, drag-and-drop the HyperMesh (`.hm`) file `link.hm` into the work area.



   b. In the **Solver input file** column, enter `link.inp`. This is the name of the solver input file HyperStudy writes during any evaluation.
   c. In the **Solver execution script** column, select ***Abaqus***.

d. In the **Solver input arguments** column enter, `$filebasename`.

	Active	Label	Varname	Model Type	Resource	Solver input file	Solver execution script	Solver input arguments
1	☑	Model 1	m_1	HyperMesh	C:/HS-1550/link.hm	link.inp	Abaqus (Abaqus)	$filebasename

Optional. In addition, you may need to edit the Abaqus environment file (ex: <ABAQUS INSTALL>\v6.11\6.11-1\site\abaqus v6.env) to include:

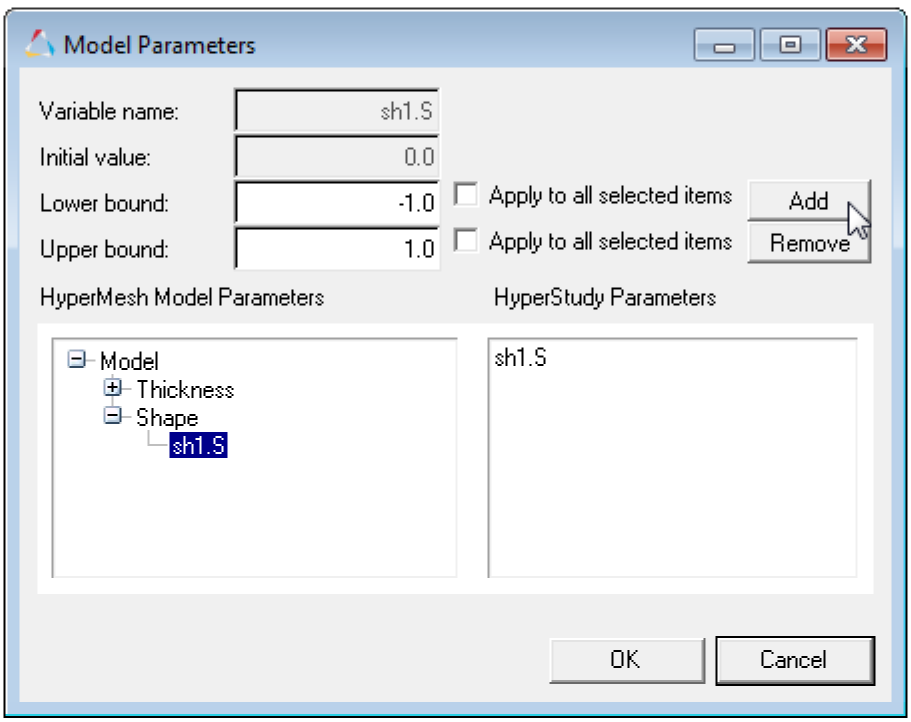
ask_delete=OFF

or

comment the line ask delete=on if any.

This is needed because Abaqus prompts you to overwrite the old files when re-running the analysis. In order to eliminate the need for user interaction, you need to command Abaqus not to ask this question and overwrite.

5. Click ***Import Variables***.
6. In the **Model Parameters** dialog, select parameters to import into HyperStudy.
   a. Expand **Shape**, and click ***sh1.S***.
   b. Verify that the **Lower bound** is `0.0` and the **Upper bound** is `1.0`.
   c. Click ***Add***.
   d. Click ***OK***.



7. Go to the **Define Input Variables** step.
8. Review the design variable's lower bound, initial and upper bound range.
9. Go to the **Specifications** step.

Step 6: Perform the Nominal Run

1. In the work area, set the **Mode** to ***Nominal Run***.
2. Click ***Apply***.
3. Go to the **Evaluate** step.
4. Click ***Evaluate Tasks***.
5. Go to the **Define Output Responses** step.

Step 7: Create and Define Output Responses

In this step you will create two output responses: Mass and Max_Stress.

1. Create the Mass output response.
   a. From the **Directory**, drag-and-drop the `link.dat` file, located in the `approaches/nom_1/run__00001/m_1` directory, into the work area.
   b. In the **File Assistant** dialog, set the **Reading technology** to ***Altair® HyperWorks®*** and click ***Next***.
   c. Select **Single item in a time series**, then click ***Next***.
   d. Define the following options, and then click ***Next***.
      - Set **Type** to ***ABAQUS.dat***.
      - Set **Request** to ***TOTAL MASS***.
      - Set **Component** to ***MASS***.
   e. Label the output response Mass.
   f. Set **Expression** to ***First Element***.
   g. Click ***Finish***. The Mass output response is added to the work area.
2. Create the Max_Stress output response.
   a. From the **Directory**, drag-and-drop the `link.obd` file, located in the `approaches/nom_1/run__00001/m_1` directory, into the work area.
   b. In the **File Assistant** dialog, set the **Reading technology** to ***Altair® HyperWorks®*** and click ***Next***.
   c. Select ***Multiple items at multiple time steps (readsim)***, then click ***Next***.
   d. Define the following options, and then click ***Next***.
      - Set **Subcase** to ***Step-2***.
      - Set **Type** to ***S-Global-Stress components (PART-1-1)***.
      - Set **Request** to ***E1*** - ***E378***.
      - Set **Component** to ***vonMises***.
   e. Label the output response Max_Stress.

f. Set **Expression** to ***Maximum***.

g. Click ***Finish***. The Max_Stress output response is added to the work area.

3. Click **Evaluate Expressions** to extract output response values.

Step 8: Run an Optimization Study

1. In the **Explorer**, right-click and select ***Add*** from the context menu.
2. In the **Add - HyperStudy** dialog, select ***Optimization*** and click ***OK***.
3. Go to the **Select Input Variables** step.
4. Review the input variable's lower and upper bound ranges.
5. Go to the **Select Output Responses** step.
6. Add an objective.

   a. Click ***Add Objective***.

   b. In the **Add - HyperStudy** dialog, add one objective.

   c. Define the objective.

   - Set **Type** to ***Minimize***.
   - Set **Apply On** to ***Mass (r_1)***.

	Active	Label	Varname	Type	Apply On	Evaluate From
1	☑	Objective 1	obj_1	Minimize	Mass (r_1)	SOLVER

7. Add a constraint.

   a. Click the ***Constraints*** tab.

   b. Click ***Add Constraint***.

   c. In the **Add - HyperStudy** dialog, add one constraint.

   d. Define the constraint.

   - Set **Apply On** to ***Max_Stress (r_2)***.
   - Set **Bound Type** to **<=** (less than or equal to).
   - For **Bound Value**, enter `200.0`.

	Active	Label	Varname	Type	Apply On	Bound Type	Bound Value
1	☑	Constraint 1	c_1	Deterministic	Max Stress (r_2)	<=	200.00000

8. Click ***Apply***.
9. Go to the **Specifications** step.
10. In the work area, set the **Mode** to ***Adaptive Response Surface Method (ARSM)***.

    Note: Only the methods that are valid for the problem formulation are enabled.

11. Click ***Apply***.
12. Go to the **Evaluate** step.
13. Click ***Evaluate Tasks*** to start the Optimization.

Step 9: View the Iteration History of the Optimization Study

1. Click the ***Iteration History*** tab to review the Optimization results. The optimal design in highlighted green.
2. Click the ***Iteration Plot*** tab to plot the Optimization results. Use the **Channel** selector to select ***Constraint 1*** and ***Objective 1***.