# Workshop - MSC Nastran Topometry Optimization with Symmetry Constraints

AN MSC NASTRAN SOL 200 TUTORIAL

Questions? Email: christian@ the-engineering-lab.com

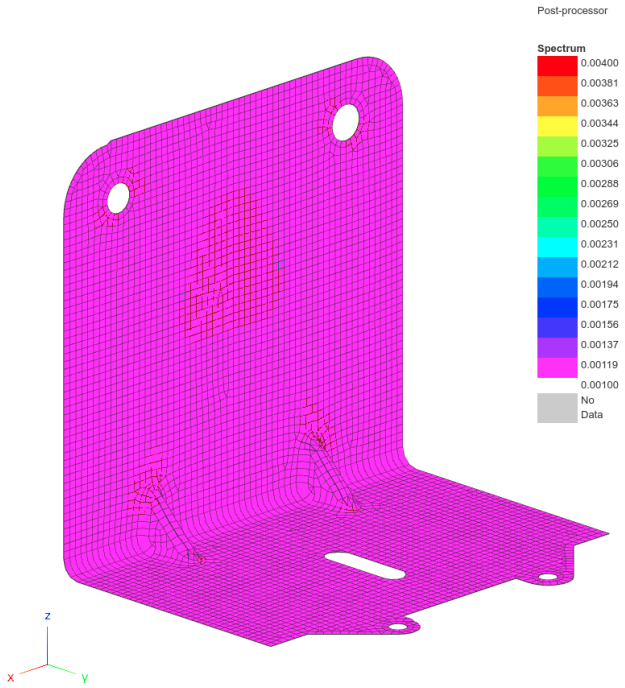
# Before Starting

**This example requires MSC Nastran 2024.1 or newer.**

# Goal: Use Nastran SOL 200 Optimization

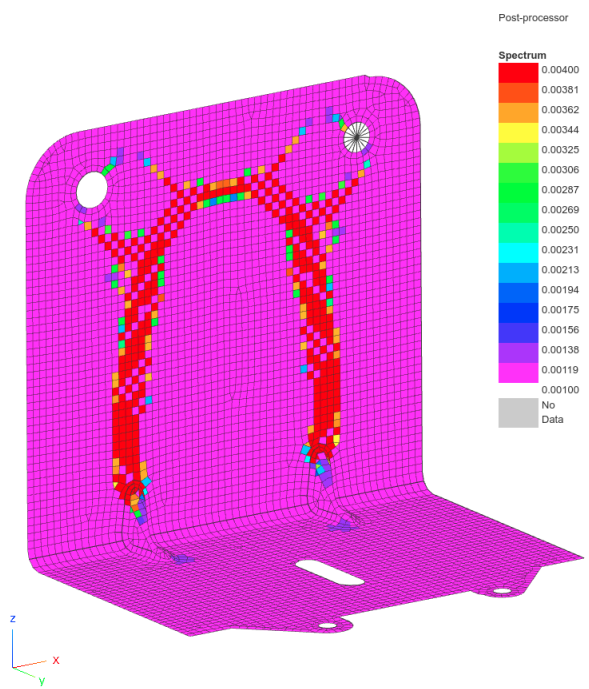
Before Optimization

- Mass: .8166 kg
- Uniform thickness

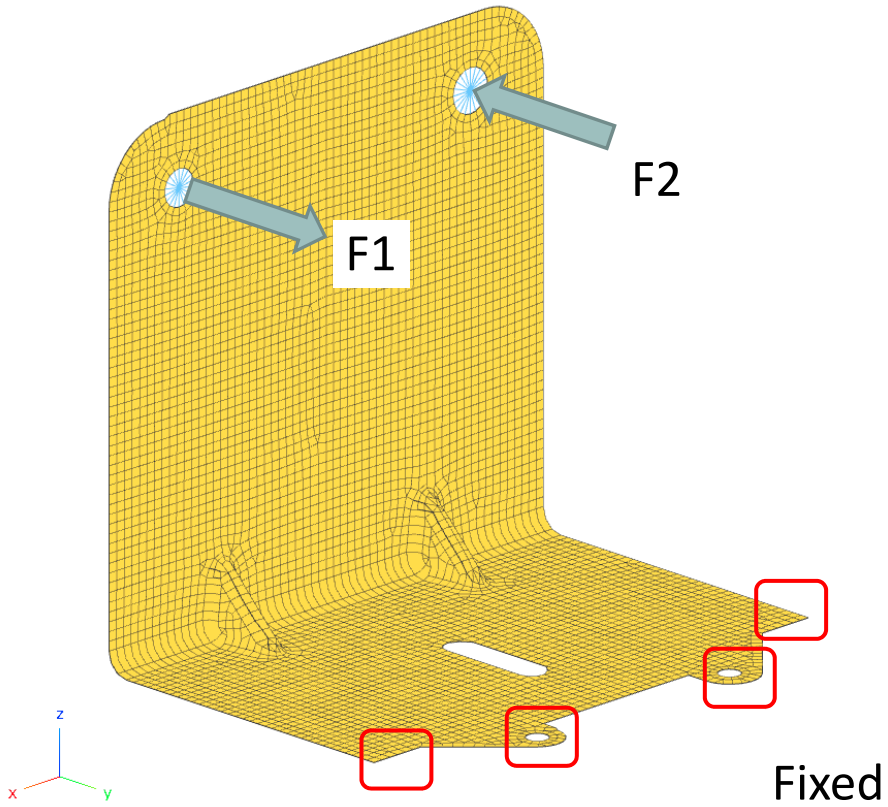



After Optimization

- Mass: .3266 kg
- Vary the thickness of each element



# Details of the structural model




Units: m, N, MPa

Material:

E: 200E9 Pa
v: .3
p: 7800 kg/m^3

Dimensions: 1m x .25m x .01m

# Optimization Problem Statement

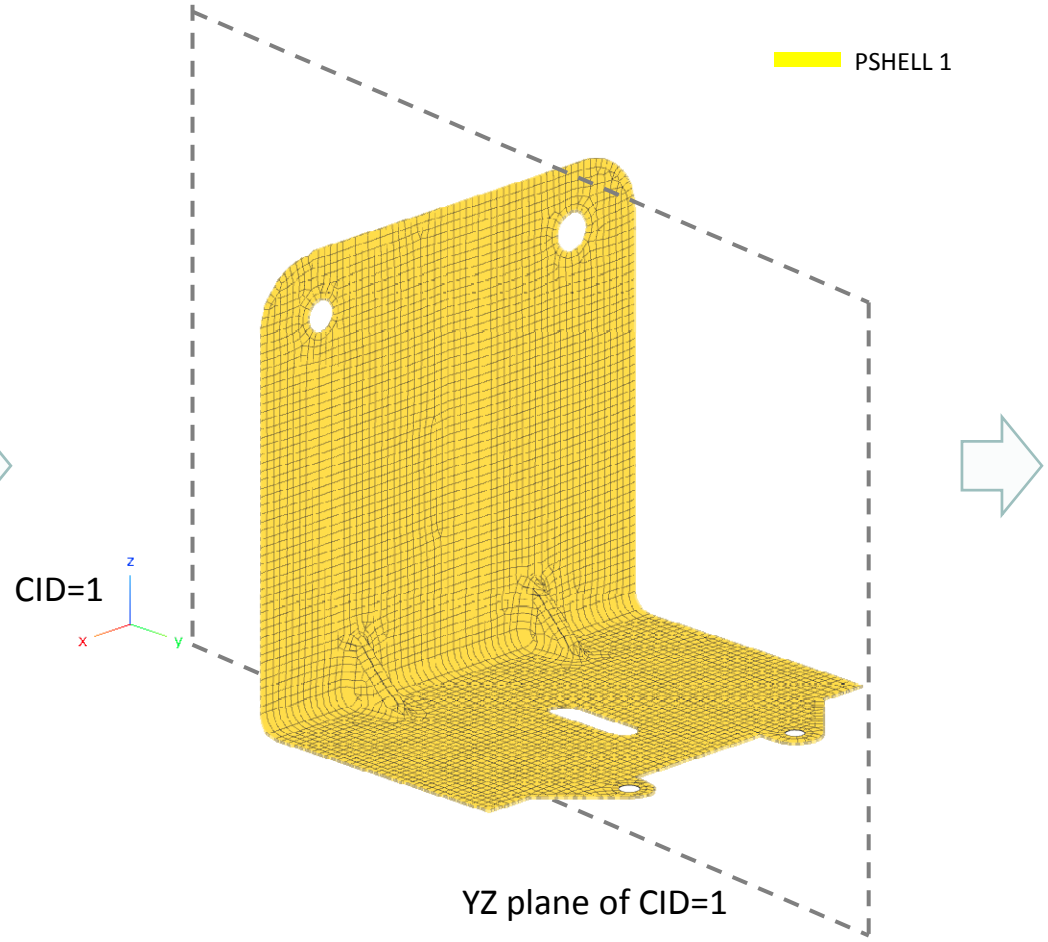
## Design Region/Variables

z1: Thickness (T) of PSHELL 1

$$.001 < z1 < .004$$

Symmetry Constraint

- Impose symmetry across the YZ plane of coordinate system 1




## Design Objective

r0: Minimize compliance

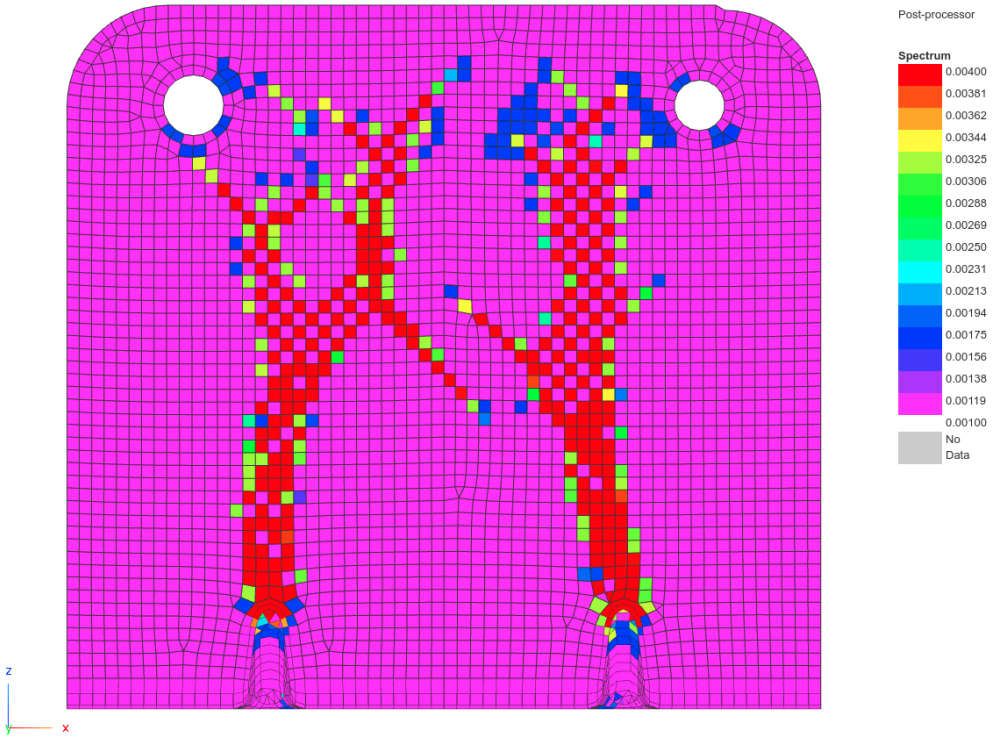
## Design Constraints

r1: fractional mass (FRMASS)

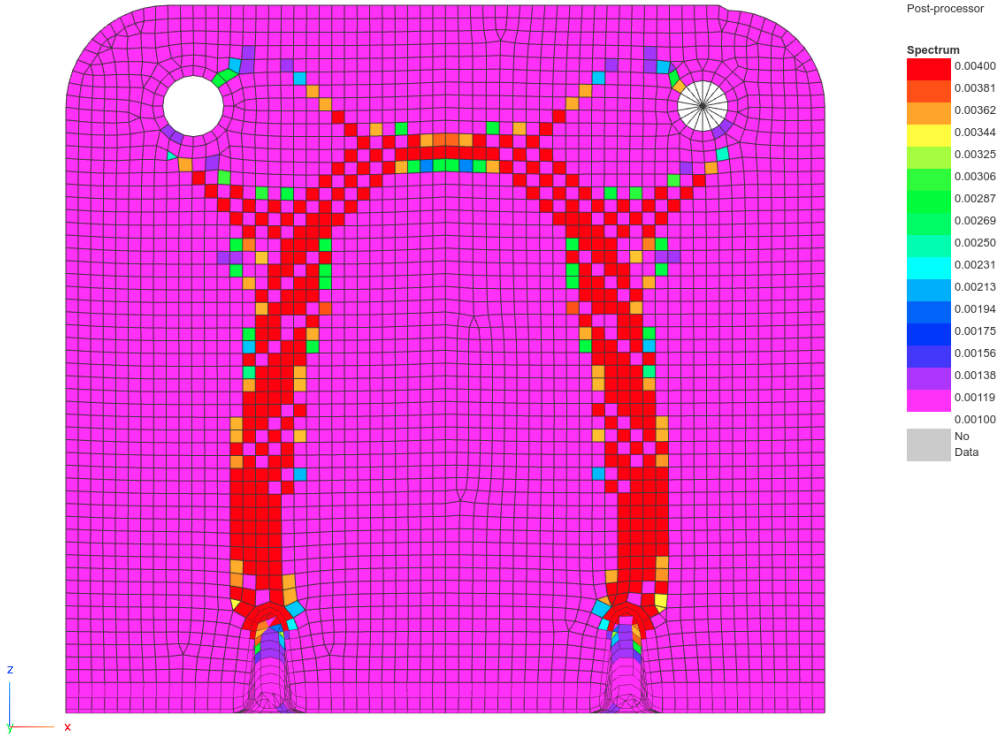
$$r1 < 0.4$$

# Comparison of Topometry Optimization With and Without Symmetry

## NO SYMMETRY




## SYMMETRY



# Contact me

- Nastran SOL 200 training
- Nastran SOL 200 questions
- Structural or mechanical optimization questions
- Access to the SOL 200 Web App

christian@ the-engineering-lab.com

# Tutorial

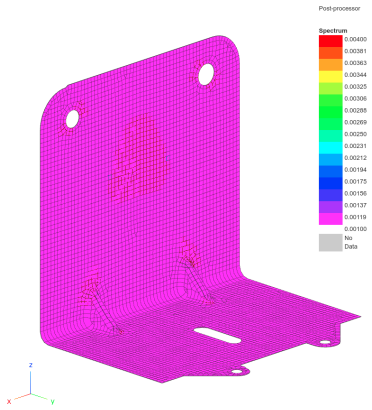
# Tutorial Overview

1. Start with a .bdf or .dat file
2. Use the SOL 200 Web App to:
   - Convert the .bdf file to SOL 200
     - Design Regions/Variables
     - Design Objective
     - Design Constraints
   - Perform optimization with Nastran SOL 200
3. Review optimization results
   - .f06
   - Topometry Optimization and Structural Results

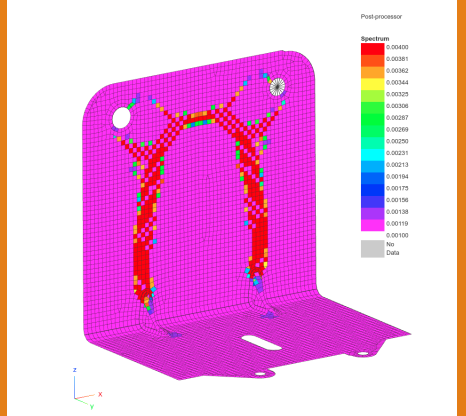
## Special Topics Covered

**Topometry Optimization with Symmetry Constraints –** Topometry optimization may lead to unsymmetric results. This tutorials discusses the use of mirror symmetry constraints to obtain symmetric results.

Before



After



# SOL 200 Web App Capabilities

The Post-processor Web App and HDF5 Explorer are free to MSC Nastran users.

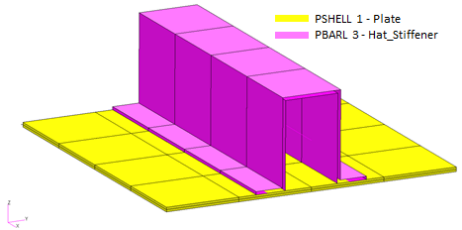
## Compatibility

- Google Chrome, Mozilla Firefox or Microsoft Edge
- Windows and Red Hat Linux
- Installable on a company laptop, workstation or server. All data remains within your company.

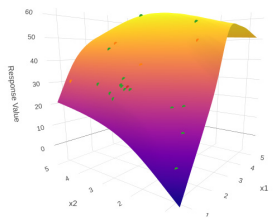
## Benefits

- REAL TIME error detection. 200+ error validations.
- REALT TIME creation of bulk data entries.
- Web browser accessible
- Free Post-processor web apps
- +80 tutorials

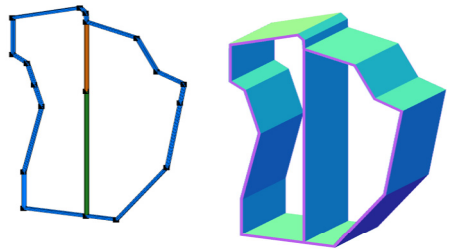
## Web Apps



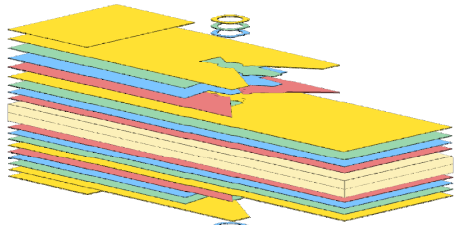

**Web Apps for MSC Nastran SOL 200**
Pre/post for MSC Nastran SOL 200. Support for size, topology, topometry, topography, multi-model optimization.



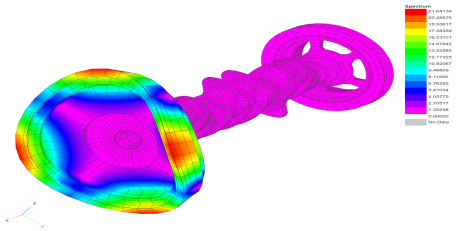
**Machine Learning Web App**
Bayesian Optimization for nonlinear response optimization (SOL 400)



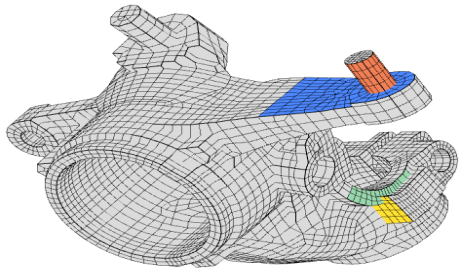
**PBMSECT Web App**
Generate PBMSECT and PBRSECT entries graphically



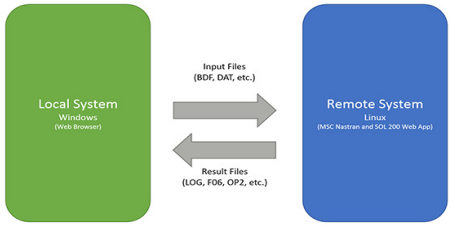
**Ply Shape Optimization Web App**
Optimize composite ply drop-off locations, and generate new PCOMPG entries



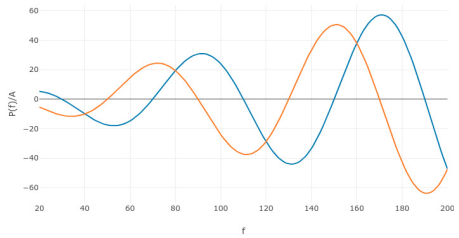
**Post-processor Web App**
View MSC Nastran results in a web browser on Windows and Linux



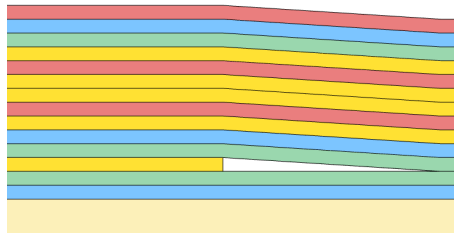
**Shape Optimization Web App**
Use a web application to configure and perform shape optimization.




**Remote Execution Web App**
Run MSC Nastran jobs on remote Linux or Windows systems available on the local network



**Dynamic Loads Web App**
Generate RLOAD1, RLOAD2 and DLOAD entries graphically



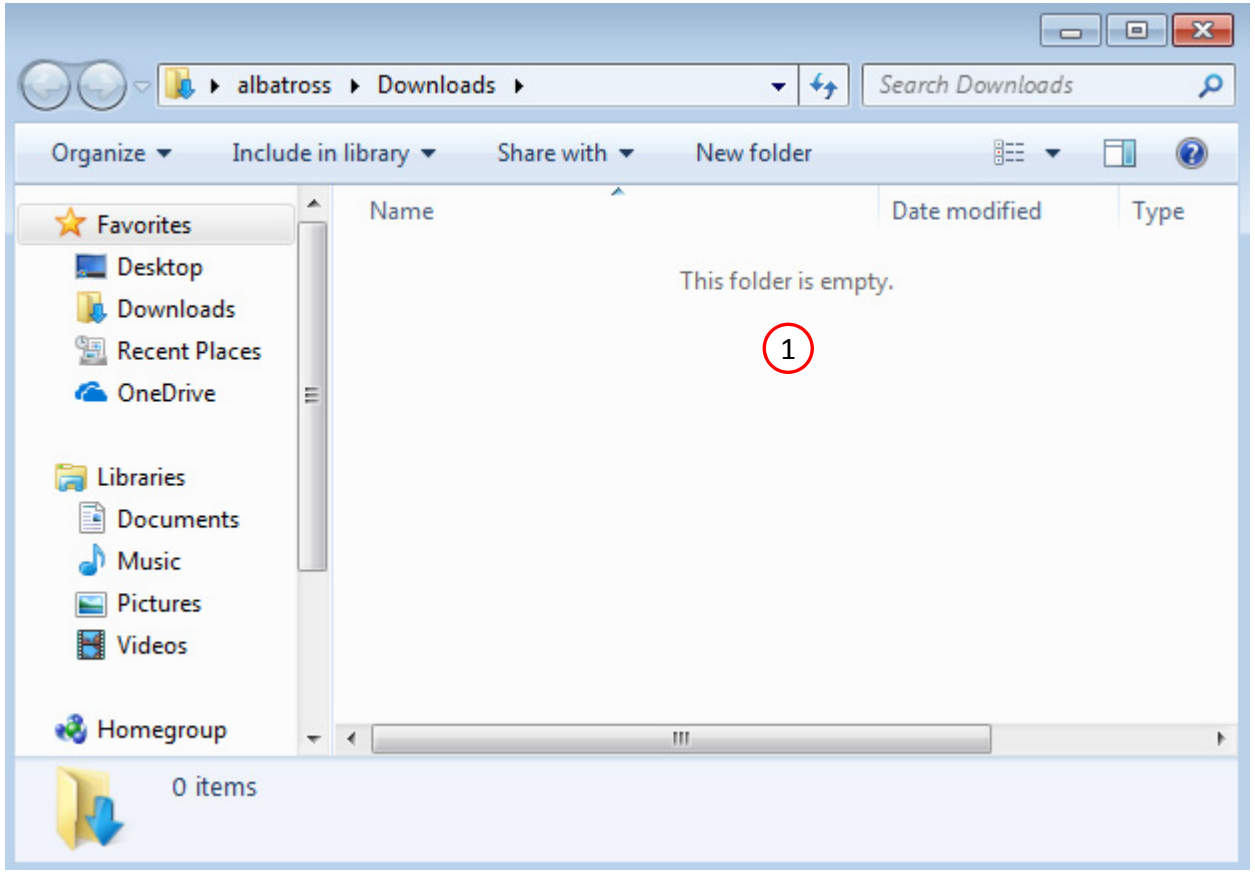
**Stacking Sequence Web App**
Optimize the stacking sequence of composite laminate plies

**HDF5 Explorer Web App**
Create graphs (XY plots) using data from the H5 file

# Before Starting

1. Ensure the Downloads directory is empty in order to prevent confusion with other files

- Throughout this workshop, you will be working with multiple file types and directories such as:
  - .bdf/.dat
  - nastran_working_directory
  - .f06, .log, .pch, .h5, etc.
- To minimize confusion with files and folders, it is encouraged to start with a clean directory.



# Go to the User's Guide

1. Click on the indicated link

- The necessary BDF files for this tutorial are available in the Tutorials section of the User's Guide.

# Obtain Starting Files

1. Find the indicated example
2. Click Link
3. The starting file has been downloaded

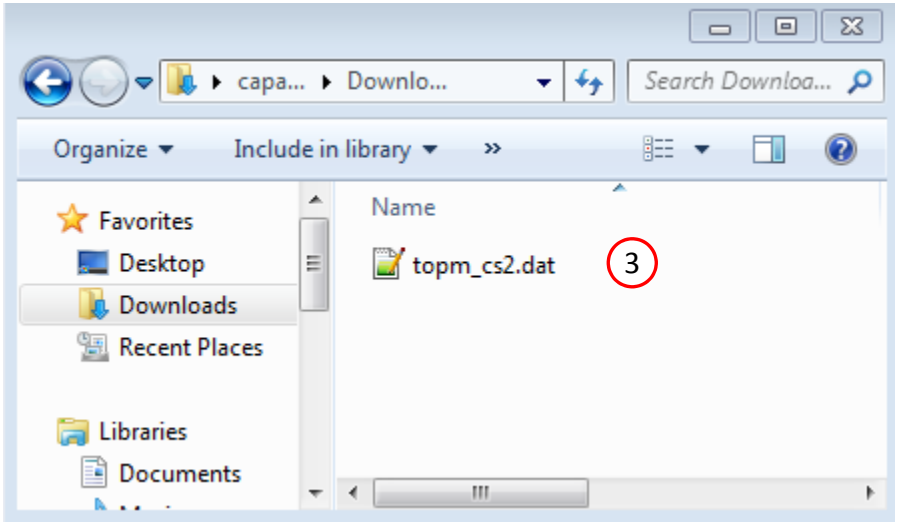
- When starting the procedure, all the necessary BDF files must be collected together.

1

**MSC Nastran Topometry Optimization with Symmetry Constraints**

This tutorial details the configuration of symmetry constraints in a topometry optimization.

Starting BDF Files: Link 2
Solution BDF Files: Link



# Open the Correct Page

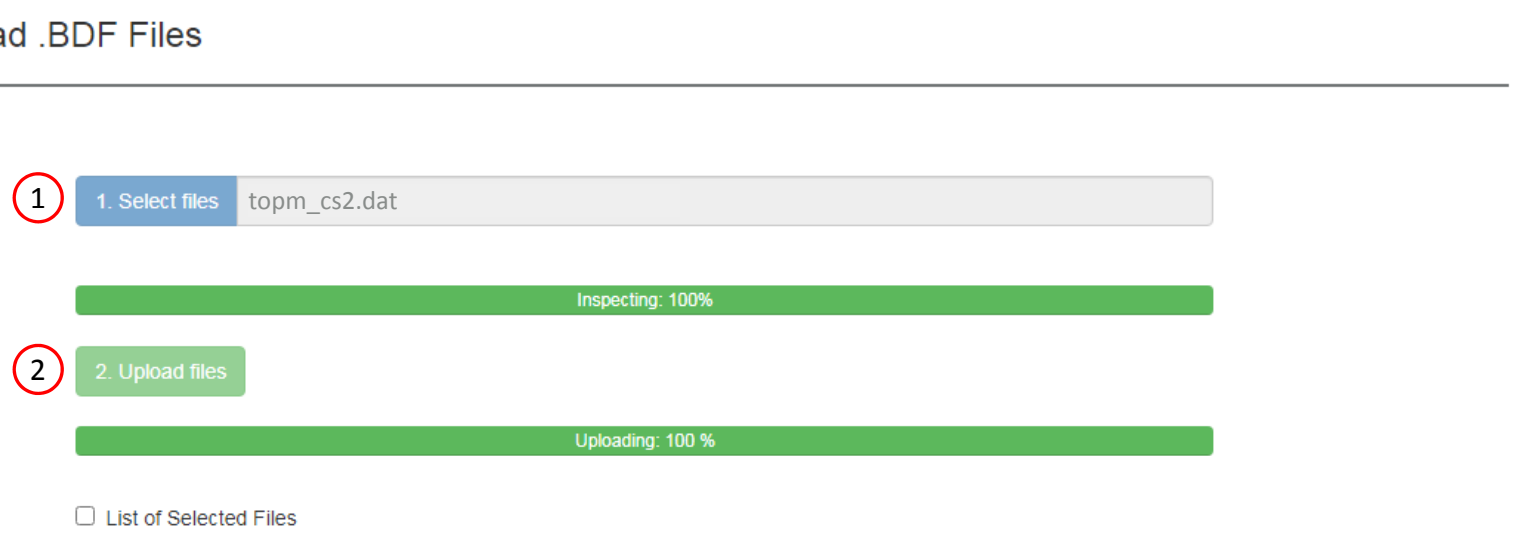
1. Click on the indicated link

- MSC Nastran can perform many optimization types. The SOL 200 Web App includes dedicated web apps for the following:
  - Optimization for SOL 200 (Size, Topology, Topometry, Topography, Local Optimization, Sensitivity Analysis and Global Optimization)
  - Multi Model Optimization
  - Machine Learning
- The web app also features the HDF5 Explorer, a web application to extract results from the H5 file type.

# Upload BDF Files

1. Click 1. Select Files and select topometry_cantilever_plate.bdf
2. Click Upload Files

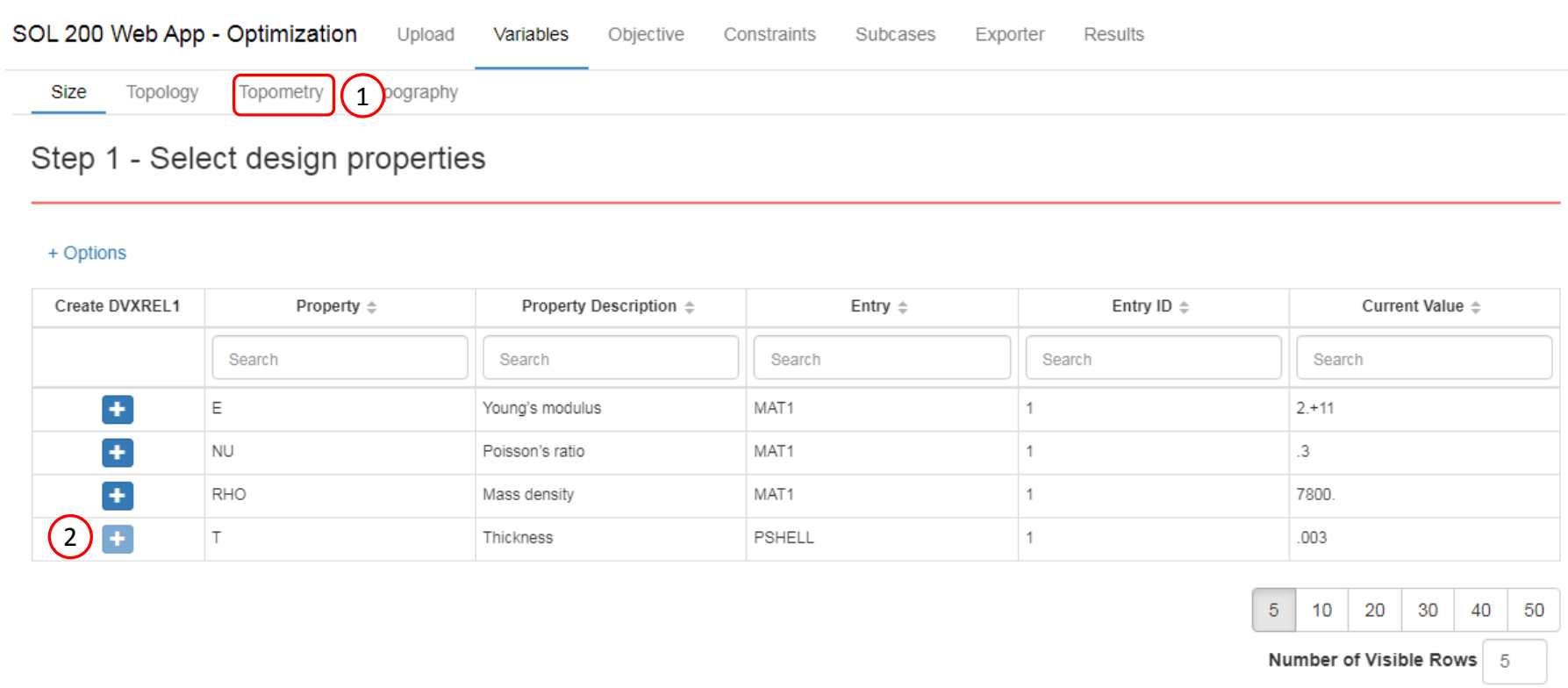
- The process starts by uploading all the necessary BDF files. The BDF files can be files of your own or files found in the Tutorials section of the User's Guide.



# Create Design Region

1. Click Topometry
2. Click on the plus (+) icon to set the thickness (T) of PSHELL 1 as a Design Region
3. The new Design Region is added to the table

- Suppose the goal is to vary the thickness. In traditional Size optimization, the thickness can be a set a single design variable. With Topometry optimization, when the design region is set, each element in the region is given its own independent thickness design variable.
- If PSHELL 1 has 500 elements associated and is configured as a design region, then there will be 500 design variables created.
- Each step has hidden functionality for advanced users. The visibility is controlled by clicking +Options .
- If the property entry, e.g. PSHELL, was given a name in Patran, e.g. Car Door, the name can be shown by marking the checkbox titled Entry Name.

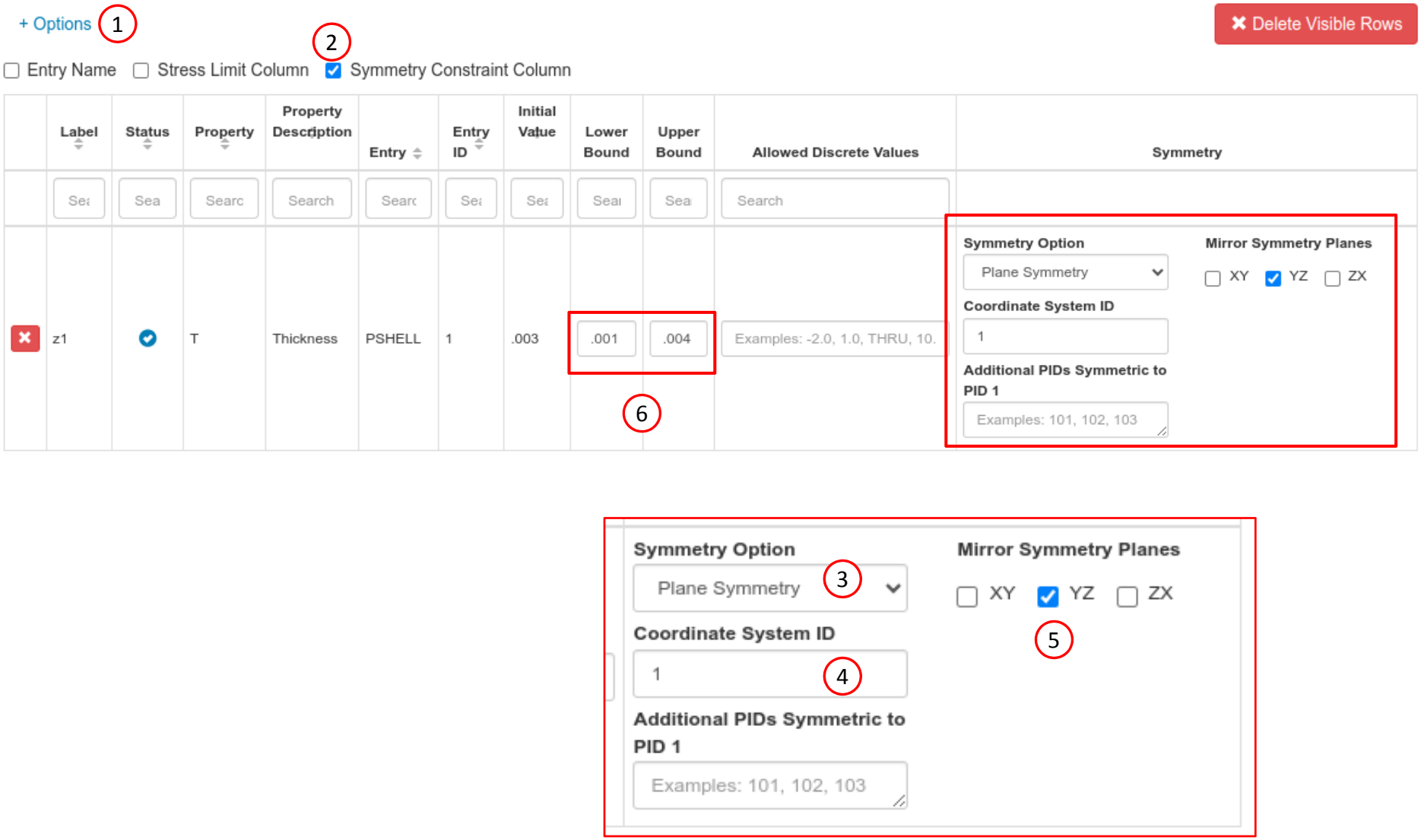


# Create Design Region

1. Click + Options
2. Mark the checkbox for Symmetry Constraint Column
3. Set Symmetry Option to Plane Symmetry
4. Set Coordinate System ID to 1
5. Set Mirror Symmetry Planes to YZ

- The defined symmetry constraints impose a requirement that the Topometry optimization solution is symmetric across the YZ plane for coordinate system 1

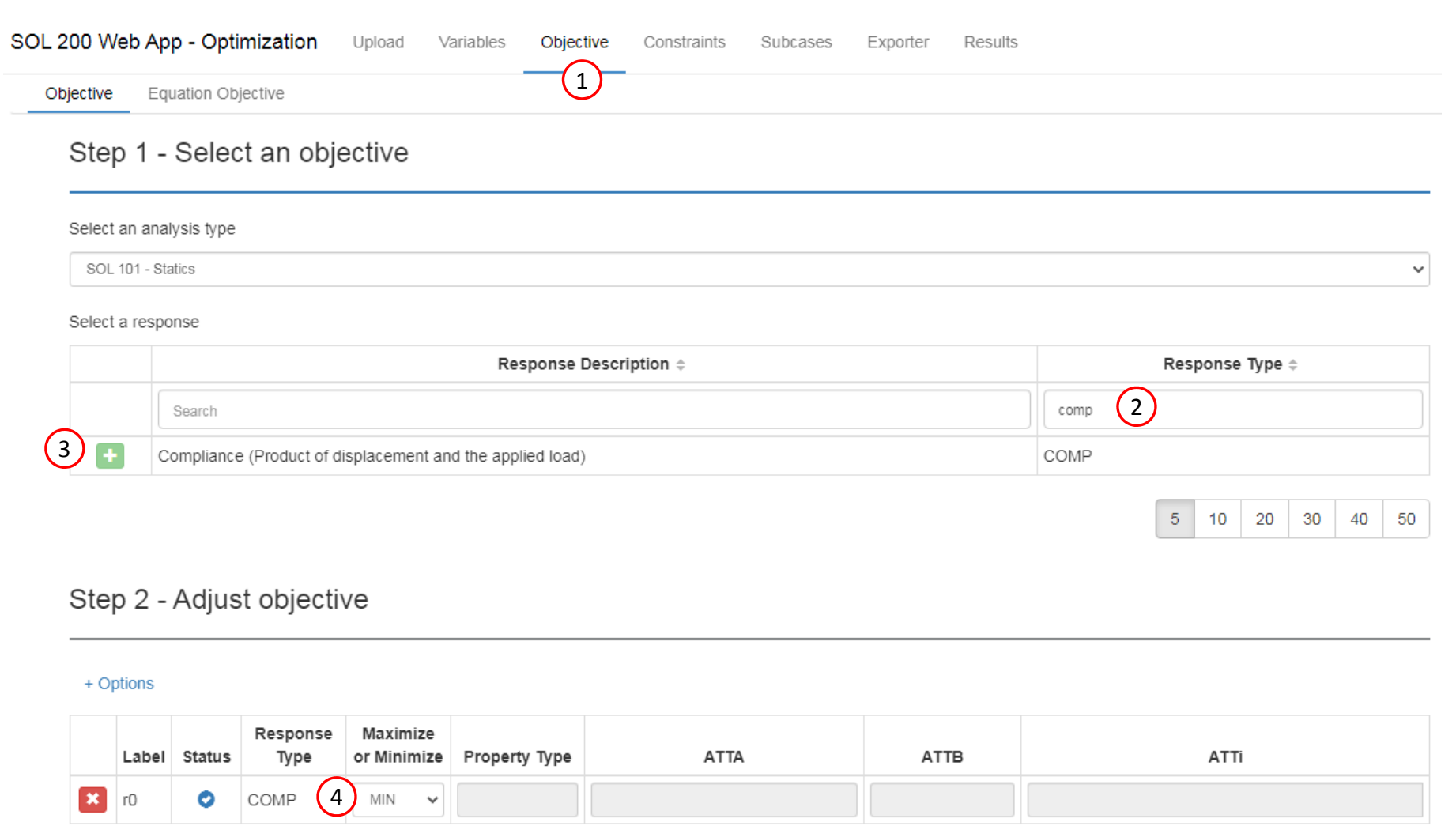
## Step 2 - Adjust TOMVAR Entries



# Create Design Objective

1. Click on Objective
2. Type ‘comp’ in the search box
3. Select the plus(+) icon for Compliance
4. The objective with label r0 is created. The objective is to minimize (MIN)

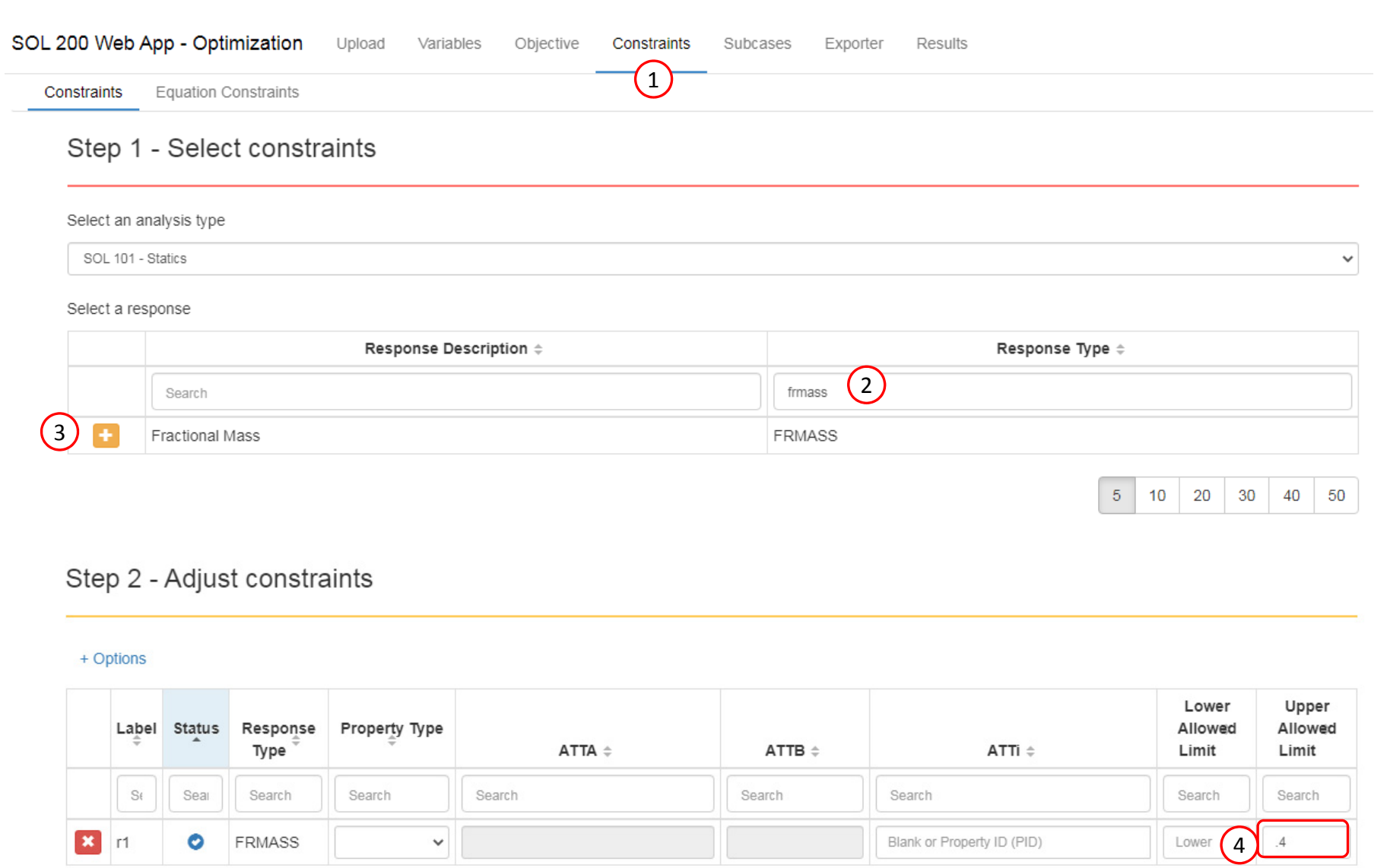
- Compliance is equal to twice the total strain energy. By minimizing the compliance/strain energy, the stiffness of the model is being maximized. See the appendix for additional details regarding compliance.



# Create Design Constraints

1. Click Constraints
2. Type ‘frmass’ in the search box
3. Select the plus(+) icon for Fractional Mass
4. Configure the following for r1
   - Upper Allowed Limit: .4
     - (Retain 40% of the material / 60% mass reduction)

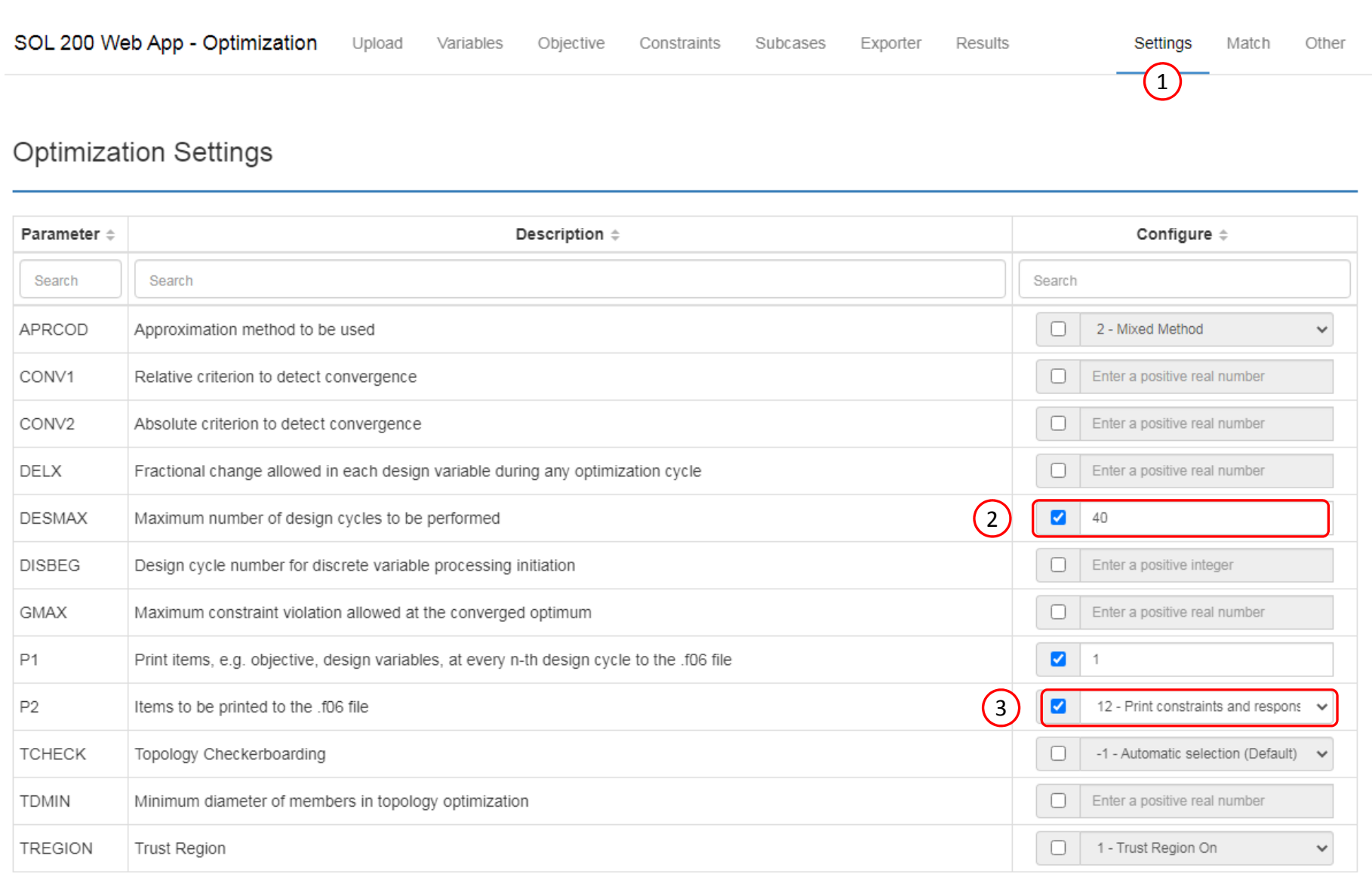
- The fractional mass constraint r1 is set for a target of .4. The optimizer will vary the design variables, normalized material densities, to produce a design that is less than or equal to 40% of the original mass.



# Configure Optimization Settings

1. Click Settings
2. Set DESMAX to 40
3. Set P2 to 12 – Print constraints and responses

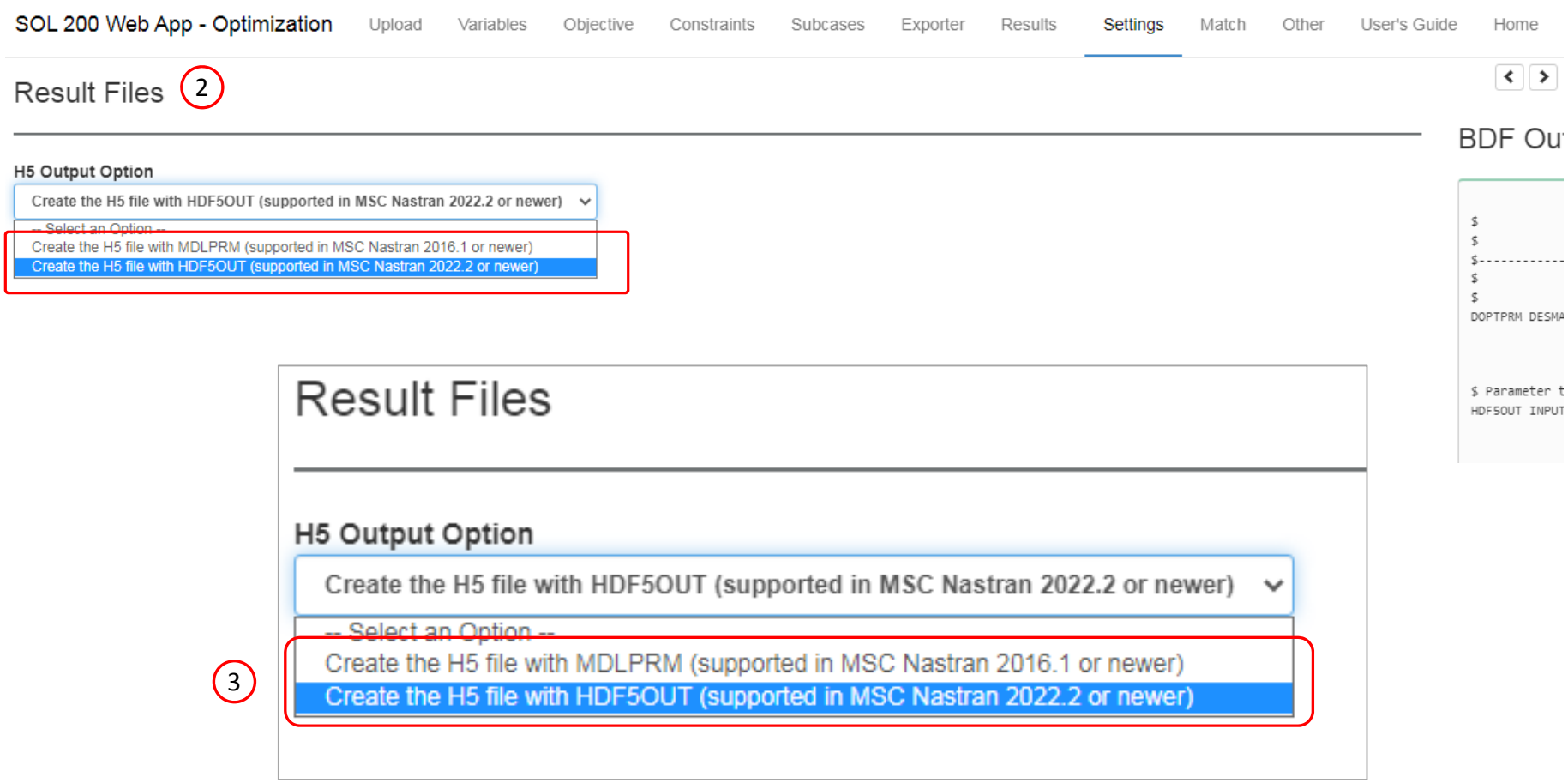
- The P2 setting controls the output of the following information to the F06 file: objective, constraints, responses, properties and design variables.
- This is a topometry optimization and will generate a large amount of property and design variable data in the F06 file. To make the F06 file size manageable, the design variable information is omitted by using the P2=12 option. When the results are viewed, note that the objective and constraint information is plotted, but the design variable history is not plotted due to the P2=12 option.
- If this is a combined size and topometry optimization, P2 should be set to 15. If this is a pure size optimization, P2 should be set to 15.



# Configure Settings

1. Scroll to section Result Files
2. Select one of the following H5 output options
   - Create the H5 file with MDLPRM
   - Create the H5 file with HDF5OUT

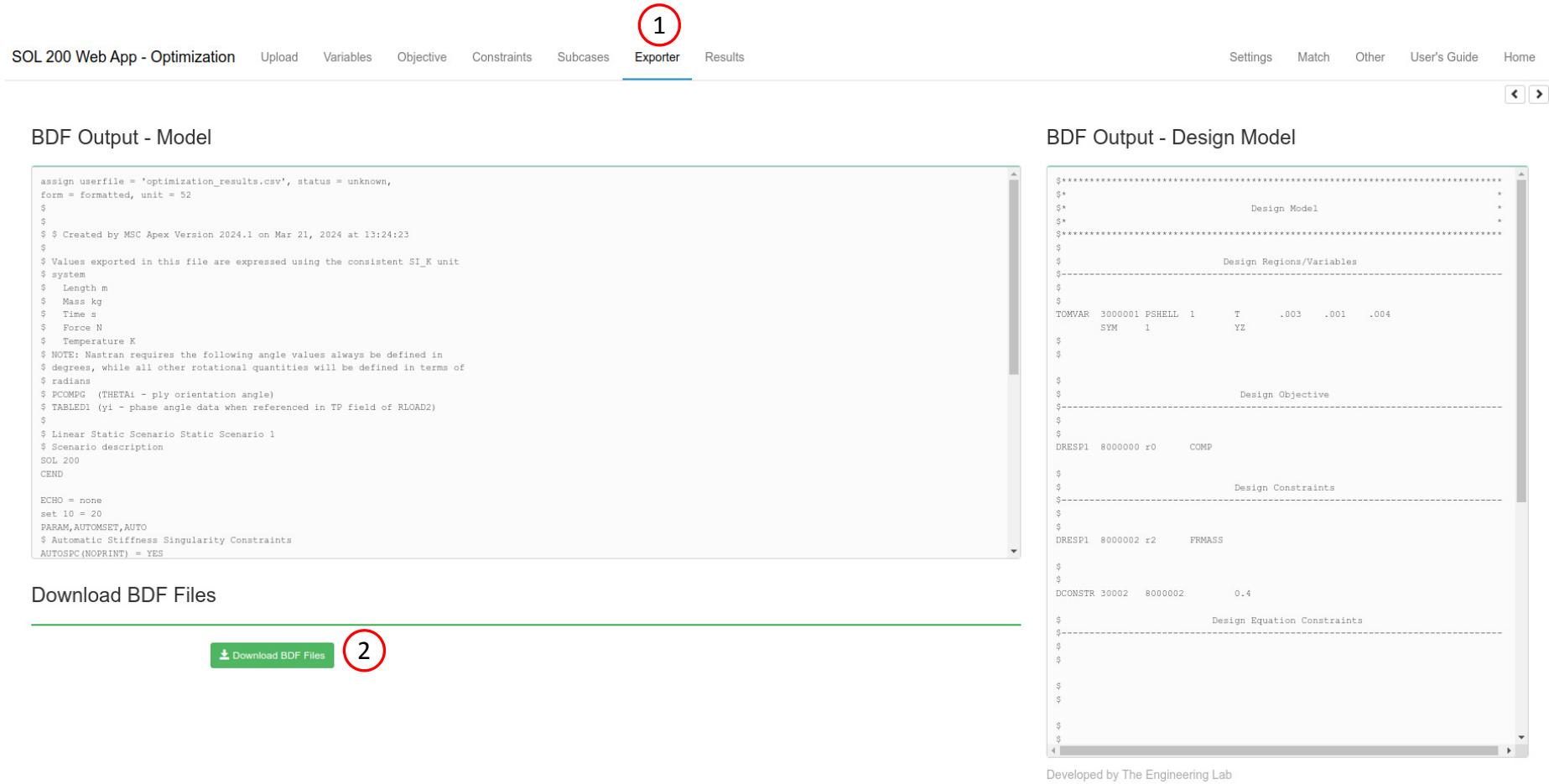
- The H5 file is used by the Post-processor web app to display MSC Nastran results.
- The H5 file is used by the HDF5 Explorer to create graphs (XY Plots) of MSC Nastran results.



# Export New BDF Files

1. Click on Exporter
2. Click on Download BDF Files

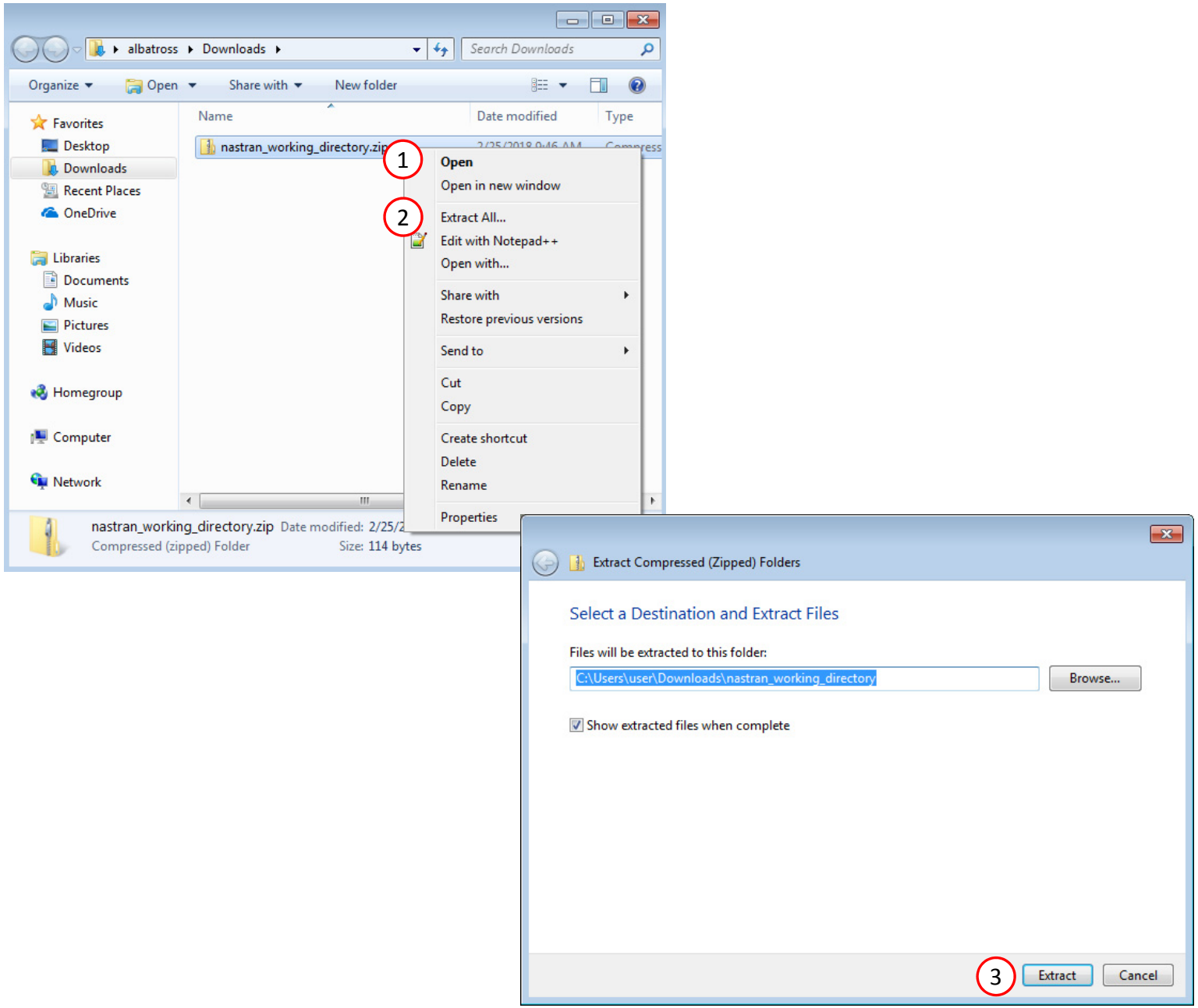
- When the download button is clicked a new file named “nastran_working_directory” is downloaded. If the file already exists in your local folder, the folder name is appended with a number, e.g. “nastran_working_directory (1).zip”



# Perform the Optimization with Nastran SOL 200

1. A new .zip file has been downloaded
2. Right click on the file
3. Click Extract All
4. Click Extract on the following window

- Always extract the contents of the ZIP file to a new, empty folder.



# Perform the Optimization with Nastran SOL 200

1. Inside of the new folder, double click on Start MSC Nastran
2. Click Open, Run or Allow Access on any subsequent windows
3. MSC Nastran will now start

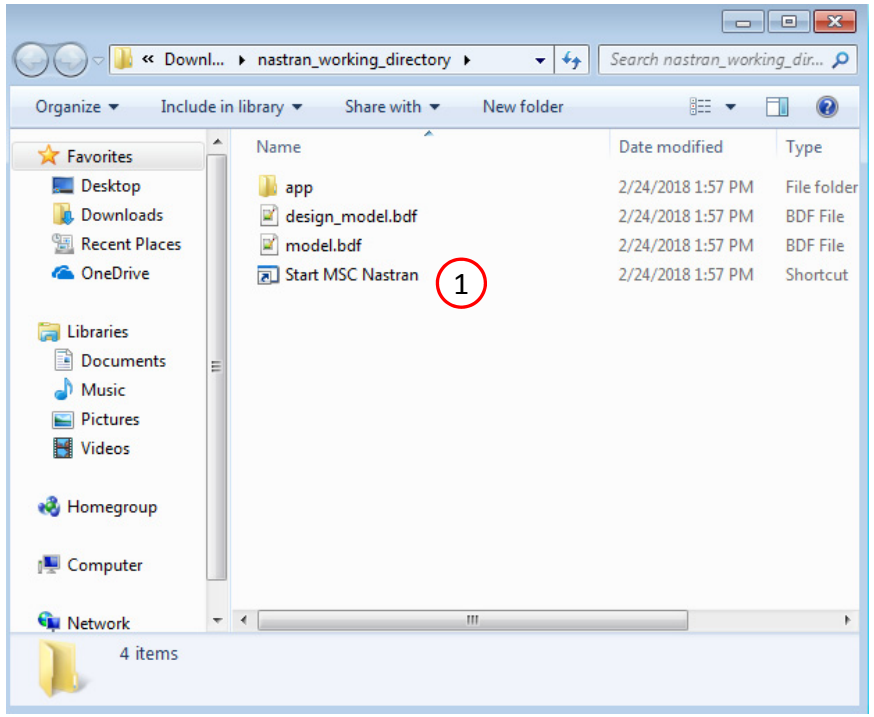
- After a successful optimization, the results will be automatically displayed as long as the following files are present: BDF, F06 and LOG.
- One can run the Nastran job on a remote machine as follows:
  1) Copy the BDF files and the INCLUDE files to a remote machine. 2) Run the MSC Nastran job on the remote machine. 3) After completion, copy the BDF, F06, LOG, H5 files to the local machine. 4) Click "Start MSC Nastran" to display the results.

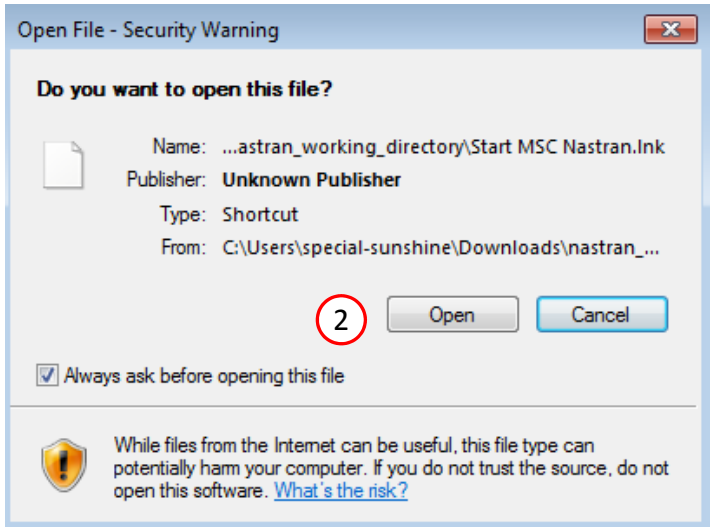
**Using Linux?**

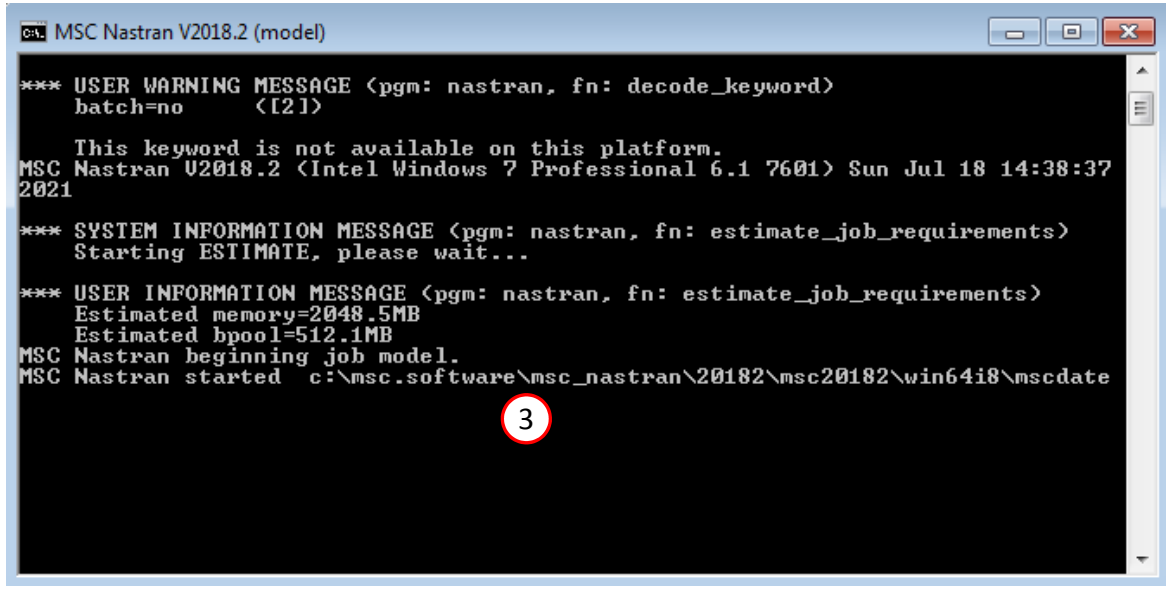
Follow these instructions:
1) Open Terminal
2) Navigate to the nastran_working_directory
cd ./nastran_working_directory
3) Use this command to start the process
./Start_MSC_Nastran.sh

In some instances, execute permission must be granted to the directory. Use this command. This command assumes you are one folder level up.

sudo chmod -R u+x ./nastran_working_directory



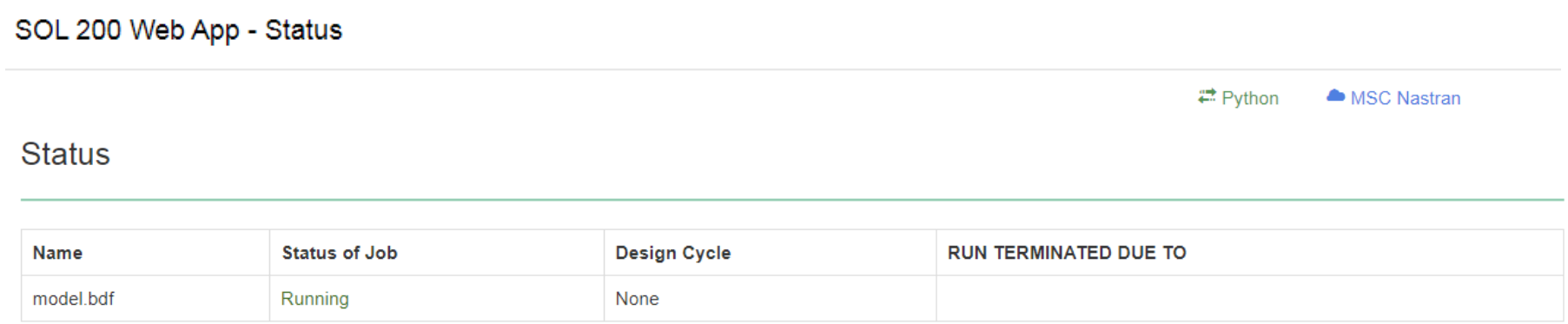



# Status

1. While MSC Nastran is running, a status page will show the current state of MSC Nastran

- The status of the MSC Nastran job is reported on the Status page. Note that Windows 7 users will experience a delay in the status updates. All other users of Windows 10 and Red Hat Linux will see immediate status updates.



# Review Optimization Results

After MSC Nastran is finished, the results will be automatically uploaded.

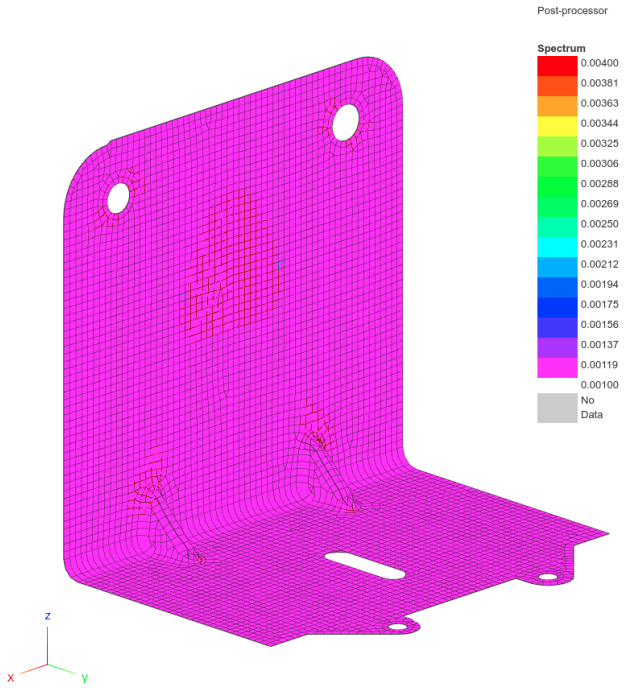
1. Ensure the messages shown have green checkmarks. This is indication of success. Any red icons indicate challenges.
2. The final value of objective and normalized constraints can be reviewed.

- Note that in a Topometry optimization, hundreds or thousands of design variables can be created. In this situation, the Design Variables are not plotted and displayed. Instead, the Objective and Normalized Constraints are displayed. It is recommended that a traditional post-processor be used to review the design variable results.

# Results

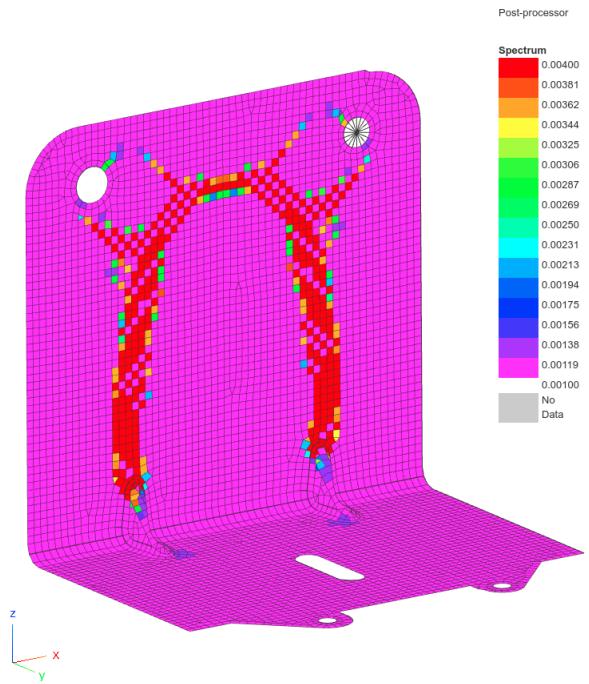
## Before Optimization

- Mass:  .8166 kg




## After Optimization

- Mass:  .3266 kg
- Vary the thickness of each element



# Update the Original Model

Ensure the BDF files prior to optimization have one of these entries:

- H5 Output
  - `MDLPRM  HDF5  1`
  - `HDF5OUT INPUT YES`

  MDLPRM HDF5 is supported in MSC Nastran 2016.1 and newer. HDF5OUT is supported in MSC Nastran 2022.2 and newer.

The following applies to MSC Nastran 2023.4 and older. For MSC Nastran 2024.1, this is not needed.

Change DESPCH1 to 1

- DESPCH
  - Before:

    `PARAM  DESPCH1 -1`
  - After:

    `PARAM  DESPCH1 1`

  DESPCH1 -1 outputs entries to the PCH file in the small field format. Since the PSHELL IDs are longer than 8 characters, the IDs appear as asterisk characters, e.g. *******. DESPCH1 1 outputs the entries in the large field format, so the IDs are fully visible.

model.pch when DESPCH=-1

```
$
$          ************************************************************
$          *                                                          *
$          *      CONTINUOUS  DESIGN  CYCLE  NUMBER  =                *
$          *                                                          *
$          ************************************************************
$
$
$          ************************************************************
$          *                                                          *
$          *      CONTINUOUS  DESIGN  CYCLE  NUMBER  =                *
$          *                                                          *
$          ************************************************************
$
$
$   UPDATED ANALYSIS MODEL DATA ENTRIES
$
PSHELL  ********      1 .001          1   1.0         1 .833333   0.0
                                0
PSHELL  ********      1 .003932       1   1.0         1 .833333   0.0
                                0
PSHELL  ********      1 .003583       1   1.0         1 .833333   0.0
                                0
PSHELL  ********      1 .00326        1   1.0         1 .833333   0.0
                                0
PSHELL  ********      1 .003114       1   1.0         1 .833333   0.0
                                0
PSHELL  ********      1 .002941       1   1.0         1 .833333   0.0
                                0
PSHELL  ********      1 .00274        1   1.0         1 .833333   0.0
                                0
PSHELL  ********      1 .002506       1   1.0         1 .833333   0.0
                                0
PSHELL  ********      1 .002254       1   1.0         1 .833333   0.0
                                0
PSHELL  ********      1 .001968       1   1.0         1 .833333   0.0
                                0
PSHELL  ********      1 .001628       1   1.0         1 .833333   0.0
                                0
```

model.pch when DESPCH=1

```
$
$          ************************************************************
$          *                                                          *
$          *      CONTINUOUS  DESIGN  CYCLE  NUMBER  =                *
$          *                                                          *
$          ************************************************************
$
$
$          ************************************************************
$          *                                                          *
$          *      CONTINUOUS  DESIGN  CYCLE  NUMBER  =                *
$          *                                                          *
$          ************************************************************
$
$
$   UPDATED ANALYSIS MODEL DATA ENTRIES
$
PSHELL* 1000000001          1  1.00007309E-03              1*
*     1.00000000E+00        1  8.33333333E-01  0.00000000E+00*
*                                           0              *
*
PSHELL* 1000000002          1  3.93152203E-03              1*
*     1.00000000E+00        1  8.33333333E-01  0.00000000E+00*
*                                           0              *
*
PSHELL* 1000000003          1  3.58257980E-03              1*
*     1.00000000E+00        1  8.33333333E-01  0.00000000E+00*
*                                           0              *
*
PSHELL* 1000000004          1  3.26027502E-03              1*
*     1.00000000E+00        1  8.33333333E-01  0.00000000E+00*
*                                           0              *
*
PSHELL* 1000000005          1  3.11369399E-03              1*
*     1.00000000E+00        1  8.33333333E-01  0.00000000E+00*
*                                           0              *
*
PSHELL* 1000000006          1  2.94087051E-03              1*
*     1.00000000E+00        1  8.33333333E-01  0.00000000E+00*
```

# Update the Original Model

The original BDF files are updated to use the new thickness distributions after a topometry optimization.

1. After a topometry optimization, new PSHELL entries are output to the PCH file.
2. Also, the 2D element entries must be updated to use the new PSHELL entry IDs.

new_2D_elements.tmp

```
CQUAD4,48368,1000000001,49013,49014,50013,50014
CQUAD4,48369,1000000002,49014,49015,50012,50013
CQUAD4,48370,1000000003,49015,49016,50011,50012
CQUAD4,48371,1000000004,49016,49017,50010,50011
CQUAD4,48372,1000000005,49017,49018,50009,50010
CQUAD4,48373,1000000006,49018,49019,50008,50009
CQUAD4,48374,1000000007,49019,49020,50007,50008
CQUAD4,48375,1000000008,49020,49021,50006,50007
CQUAD4,48376,1000000009,49021,49022,50005,50006
CQUAD4,48377,1000000010,49022,49023,50004,50005
CQUAD4,48378,1000000011,49023,49024,50003,50004
CQUAD4,48379,1000000012,49024,49025,50002,50003
```

Original BDF File

```
CQUAD4   48368       1          49013    49014    50013    50014
CQUAD4   48369       1          49014    49015    50012    50013
CQUAD4   48370       1          49015    49016    50011    50012
CQUAD4   48371       1          49016    49017    50010    50011
CQUAD4   48372       1          49017    49018    50009    50010
CQUAD4   48373       1          49018    49019    50008    50009
CQUAD4   48374       1          49019    49020    50007    50008
CQUAD4   48375       1          49020    49021    50006    50007
CQUAD4   48376       1          49021    49022    50005    50006
CQUAD4   48377       1          49022    49023    50004    50005
```

2

model.pch

```
$
$          ******************************************************************
$          *                                                                *
$          *      CONTINUOUS  DESIGN  CYCLE  NUMBER  =        30           *
$          *                                                                *
$          ******************************************************************
$
$
$   UPDATED ANALYSIS MODEL DATA ENTRIES
$
PSHELL*       1000000001               1  1.00044722E-03                1*
*         1.00000000E+00               1  8.33333333E-01  0.00000000E+00*
*                                                       0               *
*
PSHELL*       1000000002               1  1.00043722E-03                1*
*         1.00000000E+00               1  8.33333333E-01  0.00000000E+00*
*                                                       0               *
*
PSHELL*       1000000003               1  1.00043550E-03                1*
*         1.00000000E+00               1  8.33333333E-01  0.00000000E+00*
*                                                       0               *
*
PSHELL*       1000000004               1  1.00044315E-03                1*
*         1.00000000E+00               1  8.33333333E-01  0.00000000E+00*
*                                                       0               *
*
PSHELL*       1000000005               1  1.00044449E-03                1*
```

1

PCH

topm_cs2.dat

```
DISPLACEMENT(PLOT) = ALL
STRESS(PLOT,VONMISES,CORNER) = ALL
OLOAD(PLOT) = ALL
SPCFORCES(PLOT) = ALL
GPFORCE(PLOT) = ALL
MPCFORCES(SORT1,PLOT) = ALL
$
$ Event name: Event 1
$ Event description:
SUBCASE 1
 SUBTITLE=Event 1
 SPC = 5
 LOAD = 41
BEGIN BULK
hdf5out
$ HDF5 Results file
$MDLPRM,HDF5,1
PARAM,PRGPST,NO
PARAM,PRTMAXIM,NO
$.......2.......3.......4.......5.......6.......7.......8.......9.......0
$.......2.......3.......4.......5.......6.......7.......8.......9.......0
$.......2.......3.......4.......5.......6.......7.......8.......9.......0
CORD2R   1               0.      0.      0.      0.      0.      1.
         1.      0.      0.
$.......2.......3.......4.......5.......6.......7.......8.......9.......0
$ Parts & Assemblies contained in Assembly ApexdbY2
$
$
$       2       3       4       5       6       7       8       9       0
$
$
BCPARA  0       METHOD  SEGTOSEGNLGLUE  0
$
$ Part Part1
$
```

Original BDF File

# Update the Original Model

1. This Python script is used to automate the update the process.

```python
import h5py
import hdf5plugin  # This library is necessary when HDF5OUT is used (Approximately MSC Nastran 2021 and newer)
import re


def get_dataset_cquad4(path_of_h5_file):
    file = h5py.File(path_of_h5_file, 'r')
    dataset1 = file['/NASTRAN/INPUT/ELEMENT/CQUAD4']
    dataset_original = dataset1[...]

    list_of_objects = []

    for element in dataset_original:
        # Store the following fields EID, PID, G1, G2, G3, G4
        list_of_objects.append(
            {
                'eid': element[0],
                'pid': element[1],
                'g1': element[2][0],
                'g2': element[2][1],
                'g3': element[2][2],
                'g4': element[2][3]
            }
        )

    return list_of_objects


def read_cquad4_entries_from_h5_and_write_to_bdf(path_a, path_of_new_bdf_file):
    objects_a = get_dataset_cquad4(path_a)
    list_of_strings = []

    for element_i in objects_a:
        # Write the fields to an array/list
        # Ensure all array elements are strings so ','.join() works properly
        array_of_fields = [
            'CQUAD4',
            str(element_i['eid']),
            str(element_i['pid']),
            str(element_i['g1']),
            str(element_i['g2']),
            str(element_i['g3']),
            str(element_i['g4'])
        ]

        # Create the entry with comma delimiters, which is the free field format
        list_of_strings.append(','.join(array_of_fields))

    # Write the strings to a text file
    file = open(path_of_new_bdf_file, 'w')

    for item in list_of_strings:
        file.write(item + '\n')

    file.close()


def filter_entries_from_pch(path_of_pch_file, name_of_entry, path_of_new_bdf_file):
    # This function reads a PCH file and keeps specific entries
    # Before (PCH File):
    # PCOMP   10000001-.0105     0.0   650000.    TSAI   0.0     0.0       SYM
    #               70 1.5     80.         YES      70 .774108-65.         YES
    #               70 1.5     80.         YES      70 .774108-65.         YES
    # $ Spawned PSHELL, MAT2 entries from PCOMP    10000001
    # $   PSHELL*         10000001       110000001  9.09643308E+00       210000001*
    # $   *         1.00000000E+00               0  1.00000000E+00  0.00000000E+00*
    # After (new_entries.bdf):
    # PCOMP   10000001-.0105     0.0   650000.    TSAI   0.0     0.0       SYM
    #               70 1.5     80.         YES      70 .774108-65.         YES
    #               70 1.5     80.         YES      70 .774108-65.         YES

    file = open(path_of_pch_file, 'r')
    file_b = open(path_of_new_bdf_file, 'w')
    keep_line = False
    keep_continuation_line = False

    # Example: Suppose you only want to read PCOMP entries
    # 1   PCOMP   10000001-.0105     0.0   650000.    TSAI   0.0     0.0       SYM
    # 2                70 1.5     80.         YES      70 .774108-65.         YES
    # 3                70 1.5     80.         YES      70 .774108-65.         YES
    # 4   MAT1    101
    # 5   PSHELL*          10000001       110000001  9.09643308E+00       210000001*
    # 6   *         1.00000000E+00                0  1.00000000E+00  0.00000000E+00*
    # 7   *        -1.05000000E-02  9.08593308E+00       410000001
    # 8   $MAT2*           110000001  1.92043294E+06  2.00203393E+06 -4.72326628E+05*
    # 9   $*          2.27776275E+07  3.50255102E+04  2.62296904E+06  5.85260000E-02*
    # 10  $*          0.00000000E+00  0.00000000E+00  0.00000000E+00  0.00000000E+00*
    # 11  $*          0.00000000E+00  0.00000000E+00  0.00000000E+00  0.00000000E+00*
    # 12  $*                       0
    # 13  PCOMP   10000002-.0105     0.0   650000.    TSAI   0.0     0.0       SYM
    # 14               70 1.5     80.         YES      70 .774108-65.         YES
    # 15               70 1.5     80.         YES      70 .774108-65.         YES
    for line in file:
        if re.match(r'^' + name_of_entry, line):
            # This detects lines 1 and 13 which is the first line of the entry
            keep_line = True
        elif re.match(r'^(\*|\s+)', line) is None:
            # This detects lines 4, 5, 6, 7, 8, 9, 10, 11, 12 which are other entries not to keep
            keep_line = False
            keep_continuation_line = False

        if keep_line is True:
            if re.match(r'^(\*|\s+)', line):
                # This detects lines 2, 3, 14, 15 which are continuation lines of the entry
                keep_continuation_line = True

        if re.match(r'\$', line) is None:
            # This detects all lines, except lines 8, 9, 10, 11, 12 which are commented with $
            if keep_line is True or keep_continuation_line is True:
                # Write the line to a new file
                file_b.write(line)

    file.close()
    file_b.close()


if __name__ == '__main__':

    # Comments
    # 1. This python script outputs updated PCOMP and CQUAD4 elements after an MSC Nastran topometry optimization.
    # 2. Modify path_a and path_b, then run this script
    # 3. This works as long MDLPRM,HDF5,1 is used, which triggers the output of the
    #    INPUT datasets to the H5 file. The INPUT datasets are the bulk data entries: GRIDs, CQUAD4s, PSHELLs,
etc.

    path_a = '/home/usera/Downloads/nastran_working_directory/model.h5'
    path_b = '/home/usera/Downloads/nastran_working_directory/model.pch'

    # Output New QUAD4 Elements After Topometry Optimization
    # ###############################################################################
    # Output updated CQUAD4 entries
    read_cquad4_entries_from_h5_and_write_to_bdf(path_a, 'new_2D_elements.tmp')

    # Output
    # CQUAD4*,1,1000000001,1,2,16,15
    # CQUAD4*,2,1000000002,2,3,17,16
    # CQUAD4*,3,1000000003,3,4,18,17
    # CQUAD4*,4,1000000004,4,5,19,18
    # CQUAD4*,5,1000000005,5,6,20,19
    # CQUAD4*,6,1000000006,6,7,21,20
    # CQUAD4*,7,1000000007,7,8,22,21
    # [...]

    # Output New PCOMP Entries After Topometry Optimization
    # ###############################################################################
    filter_entries_from_pch(path_b, 'PSHELL', 'new_pshell_entries.tmp')
    # Outputs
    # PCOMP   10000001-.0105     0.0   650000.    TSAI   0.0     0.0       SYM
    #               70 1.5     80.         YES      70 .774108-65.         YES
    #               70 1.5     80.         YES      70 .774108-65.         YES
    # PCOMP   10000002-.0105     0.0   650000.    TSAI   0.0     0.0       SYM
    #               70 1.39312 80.         YES      70 .052964-65.         YES
    #               70 1.39312 80.         YES      70 .052964-65.         YES
```

# Update the Original Model

The Python script generates a new TMP file.

1. Copy and paste the CQUAD4 elements to the original BDF file.

new_2D_elements.tmp

Original BDF File

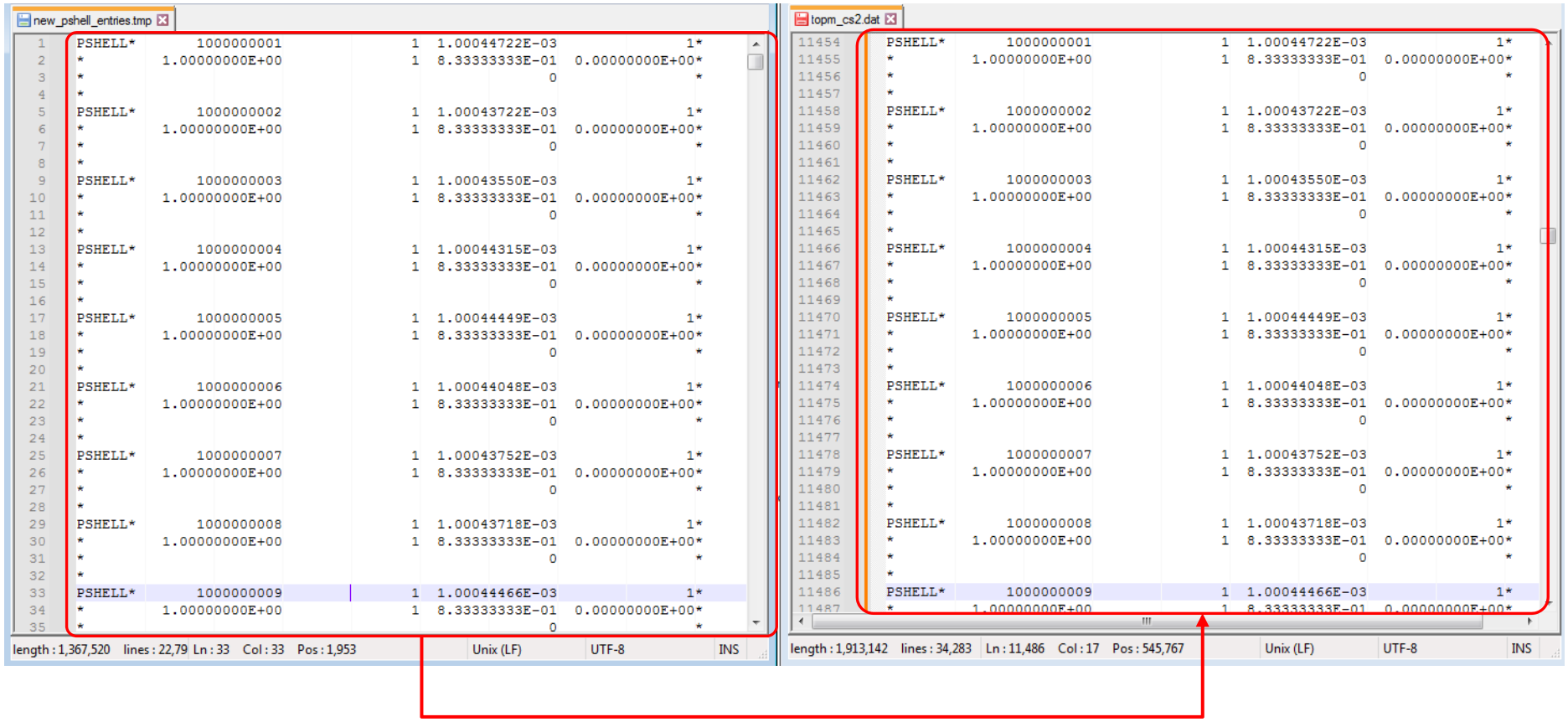
# Update the Original Model

The Python script generates a new TMP file.

1. Copy and paste the PSHELL elements to the original BDF file.

new_pshell_entries.tmp

Original BDF File




1

# Inspection of MSC Nastran Results with the Post-processor Web App

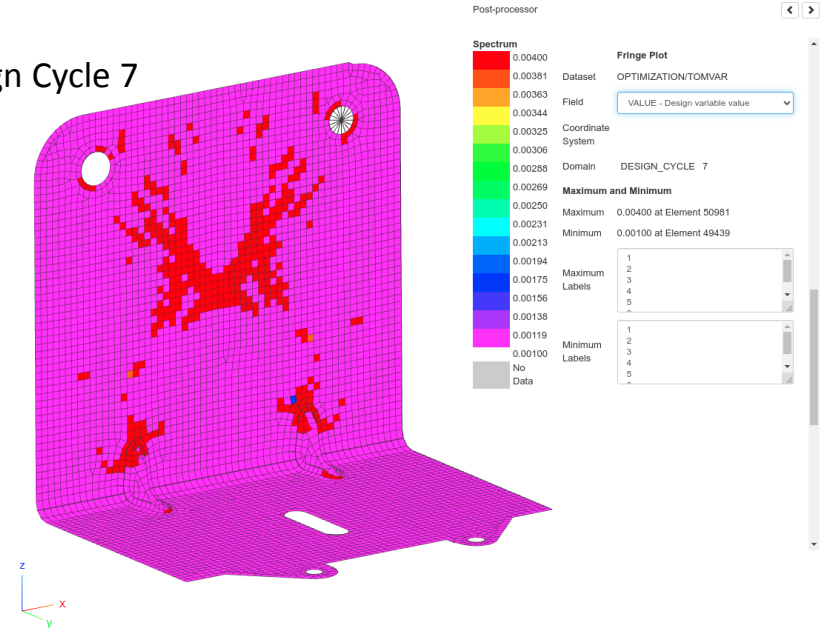
# Post-processor Web App

- The Post-processor web app is used to inspect the MSC Nastran results.
- Consider the results of the topometry optimization which are thickness distributions.

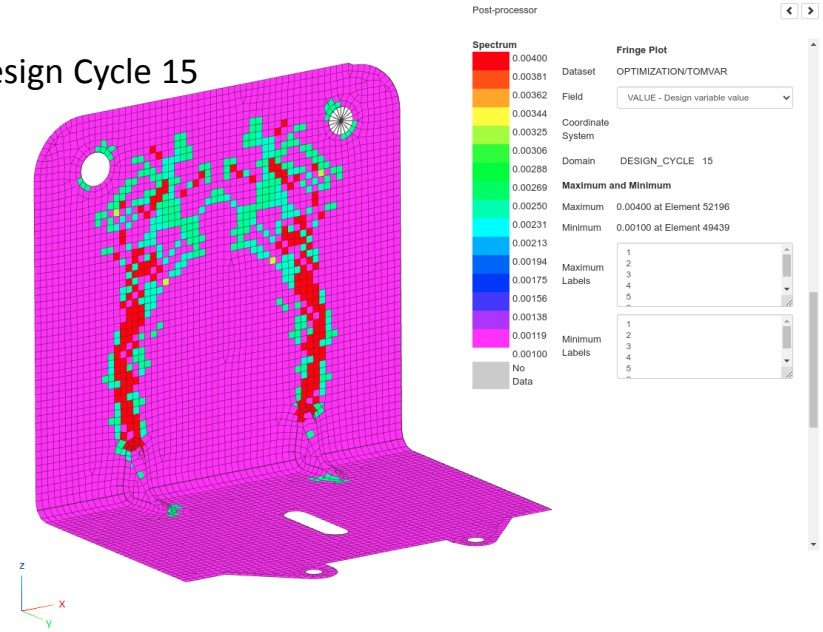
- Refer to the Post-processor web app tutorials to learn more about MSC Nastran results.

## Topometry Optimization Results – Thickness Distribution

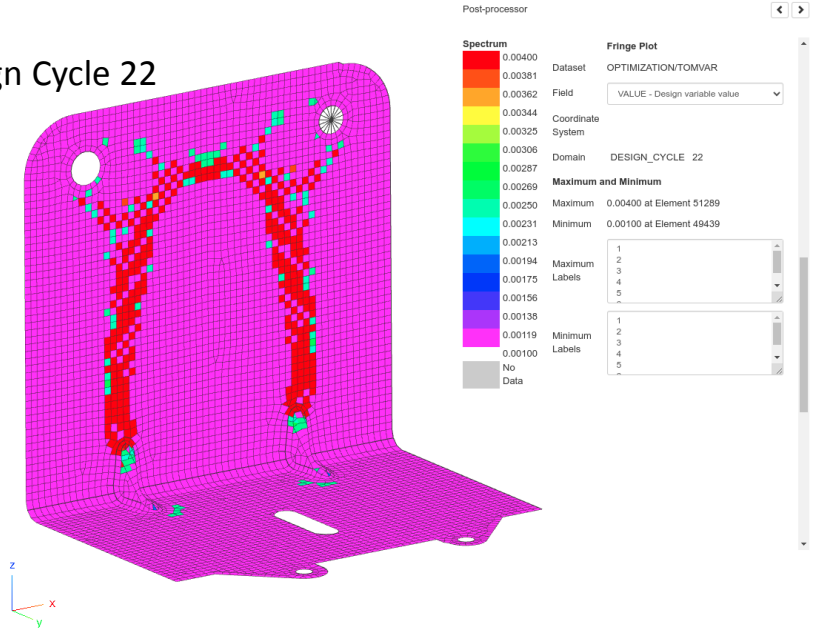
Design Cycle 7



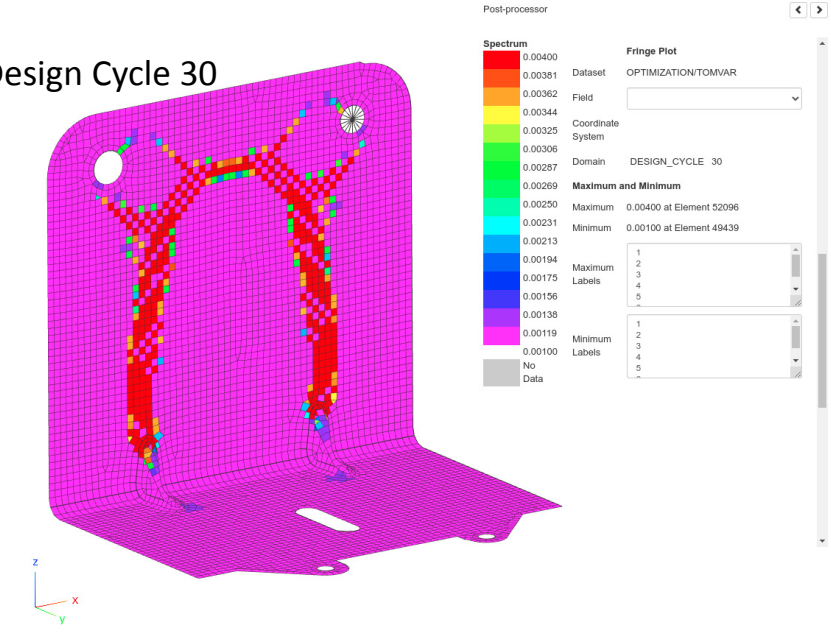

Design Cycle 15




Design Cycle 22




Design Cycle 30



# Final Comments

The SYM and STRESS keywords should not be used together and results in UFM 7052 and 7002.

```
   *** USER FATAL MESSAGE 7052 (DOMPTC)
       ILLEGAL PROPERTY TYPE  IS REFERENCED ON  ENTRY .
    *** USER FATAL MESSAGE 7002 (IFP10F)
       NO ELEMENTS ARE REFERENCED BY ANY TOMVAR PROPERTY IDs
```

NOT OK

```
    TOMVAR  3000001 PSHELL  1       T       .003    .001    .004
            STRESS  1.57E9
            SYM     1               YZ
```

OK

```
    TOMVAR  3000001 PSHELL  1       T       .003    .001    .004
            STRESS  1.57E9
    or
    TOMVAR  3000001 PSHELL  1       T       .003    .001    .004
            SYM     1               YZ
```

End of Tutorial