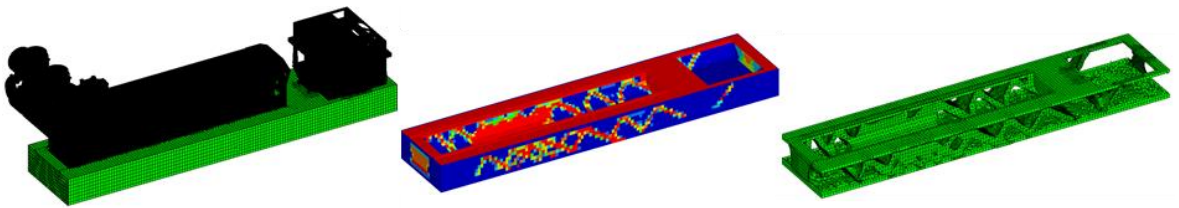


RESEARCH REPORT

VTT-R-00527-15



SIMPRO Task 2.2. Topology Optimization in HPC Environment

Authors: Erin Komi, Juhani Hämäläinen

Confidentiality: Public

Report's title		
SIMPRO/VTT Task 2.2. - Topology Optimization in HPC Environment		
Customer, contact person, address		Order reference
Tekes, Matti Säynätjoki Kyllikinportti 2, P.O. Box 69, FI-00101 Helsinki, FINLAND		Tekes: 40204/12
Project name		Project number/Short name
Topology Optimization in HPC Environment		78634 / SIMPRO
Author(s)		Pages
Erin Komi, Juhani Hämäläinen		28
Keywords		Report identification code
topology optimization, high-performance computing		VTT-R-00527-15
Summary		
<p>This study sought to investigate the use of a high-performance computing environment to efficiently automate some of the early steps in the design process. The idea was to manipulate commercially available software to automate the running and results tabulation of a large number of topology optimizations. This was achieved by creating a finite element model of the chosen case study (genset base frame), creating superelements for large, complex components that were attached (engine and generator), preparing a topology optimization model, generating shape variables that altered the length, width and height of the design space, and launching a design of experiments on a computing cluster so that 64 variations of the topology optimization were run. Upon completion of each optimization run, the DOE automatically launched a program to interpret the results in several different ways, and then subsequently submitted the new design for reanalysis based on the initial design criteria. The results of the reanalysis were automatically tabulated for easy comparison of designs. The DOE set-up was also flexible enough to allow for additional topology optimization results interpretation and reanalysis after the initial runs were completed. The final result was hundreds of designs from which it was easy to select those which were most able to achieve the design goals.</p>		
Confidentiality	Public	
Tampere 2.4.2015		
Written by	Reviewed by	Accepted by
Erin Komi Research Scientist	Kai Katajamäki Principal Scientist	Johannes Hyrynen Head of Research Area
VTT's contact address		
P.O. Box 1300, FI-33101 Tampere, Finland		
Distribution (customer and VTT)		
Customer, VTT		
<p><i>The use of the name of the VTT Technical Research Centre of Finland (VTT) in advertising or publication in part of this report is only permissible with written authorisation from the VTT Technical Research Centre of Finland.</i></p>		

Preface

The contents of this report describe work done for SIMPRO project sub-task 2.2, topology optimization in a high-performance computing (HPC) environment.

The use of topology optimization early in the design process for illustrating possible material placement has become a very attractive tool. However, for complex and detailed components, some computational challenges still exist. In this task, the topology optimization process is run in a HPC environment. The optimization is controlled by a design of experiments (DOE) process, whereby the initial design space can be varied at the start of each run, ensuring that a number of unique solutions will be found. The DOE process is parallelized, thereby reducing the time needed to achieve multiple potential design solutions.

The utilized software for this task are Altair OptiStruct, HyperMesh, HyperStudy, HyperMorph, and OSSmooth.

Tampere 2.4.2015

Authors

Contents

Preface.....	2
Contents.....	3
1. Introduction.....	4
2. Goal.....	4
3. Methods.....	5
3.1 Parallelization of OptiStruct.....	7
3.2 FEM model creation.....	8
3.3 Topology optimization.....	9
3.4 Model parameterization.....	11
3.5 Design of experiments.....	11
3.6 Mesh generation for reanalysis after topology optimization.....	12
3.7 Running simulation in HPC environment.....	13
4. Optimization results.....	13
5. Limitations.....	17
6. Conclusions.....	17
7. Suggestions for future work.....	18
References.....	18
Appendix A – Creation of shape variables with HyperMorph.....	19
Appendix B – Design of experiments set-up in HyperStudy and launch on computational cluster.....	21

1. Introduction

Finite element (FE) based topology optimization is a process of finding the optimal material placement, orientation and connectivity in a given design space, dependent on loading and boundary conditions, as well as objectives and constraints of the optimization problem. This is done, as described by the example in Figure 1, by generating a finite element mesh representing the allowable design space (shown in blue in the top two images), applying appropriate loading and boundary conditions, and then determining the optimal placement of material within that space (i.e. choosing which elements should contain material and which should be without) [1]. A well-described problem can produce very informative results which can be particularly useful early in the design process. The idea presented here is to go one step further and provide an efficient means of giving designers a number optimized topology variations that meet their requirements.

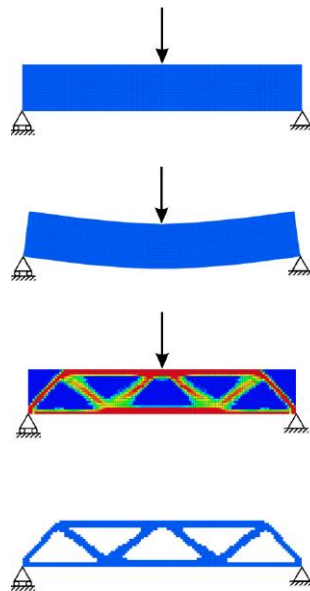


Figure 1. Example topology optimization problem – determining the optimal placement of material for achieving maximal stiffness with a prescribed volume constraint [1].

2. Goal

The goal of this task is to investigate an efficient means of producing many optimized topologies, from which designers can select the most suitable. The idea proposed is to create a design of experiments (DOE) in which the design variables are the initial dimensions of the topology optimization design space (e.g. length, width and height). Thus a large number of optimized topologies will be produced. The optimization results will automatically be interpreted, a mesh of the new design generated such that initial loads and boundary conditions remain intact, and a reanalysis of the design conducted to determine if design requirements are met. The calculations will be run in a high-performance computing (HPC) environment so that parallel computing can be exploited by two means. First parallelization will be employed by the used solver to speed up the solution time of an individual optimization run, and second, the DOE software will launch and control multiple runs simultaneously.

3. Methods

Parallelization of computational runs in a HPC environment can be expensive in terms of both number of computers being tied up during a computation and number of solver licenses needed. Therefore the first stage of this project included a pre-study on the parallelization of the used solver, OptiStruct.

In order to obtain meaningful results from this study, a challenging case was chosen to examine. The selected structure is the base frame of a diesel generating set (genset), an example of which is shown in Figure 2. A genset consists of a diesel engine connected to a generator via a flexible coupling. The engine and generator are mounted on a common base frame, which is dynamically isolated from the concrete foundation by steel springs. Generator sets produce electricity for various purposes, including off-shore facilities, ship propulsion, or as power plants.



Figure 2. Diesel generating set (Wärtsilä Power Plants).

The typical way in which this topology design process might proceed is shown in Figure 3, where the problem starts with an initial design space of which a finite element model is created. Loads and boundary conditions are defined, design and non-design space is identified and the optimization job created. The topology optimization is performed, and the results are somehow interpreted, a mesh of the results created, and a final analysis of the design is conducted to compare with initial design constraints.

A flow chart describing the DOE-driven solution process for this project is shown in Figure 4, with each box briefly explaining the tasks. The blue boxes describe the steps commonly used for topology optimization, and the orange boxes contain the additional steps used in the present study. There are two places in this process where the design is 'split', producing a number of possible final results. The first is at the stage where the model design space is parameterized, which essentially creates a number of different starting points for the topology

optimization. The second is where the optimization results are interpreted and a new mesh is created. Significantly different designs can be obtained depending on how this procedure is run.

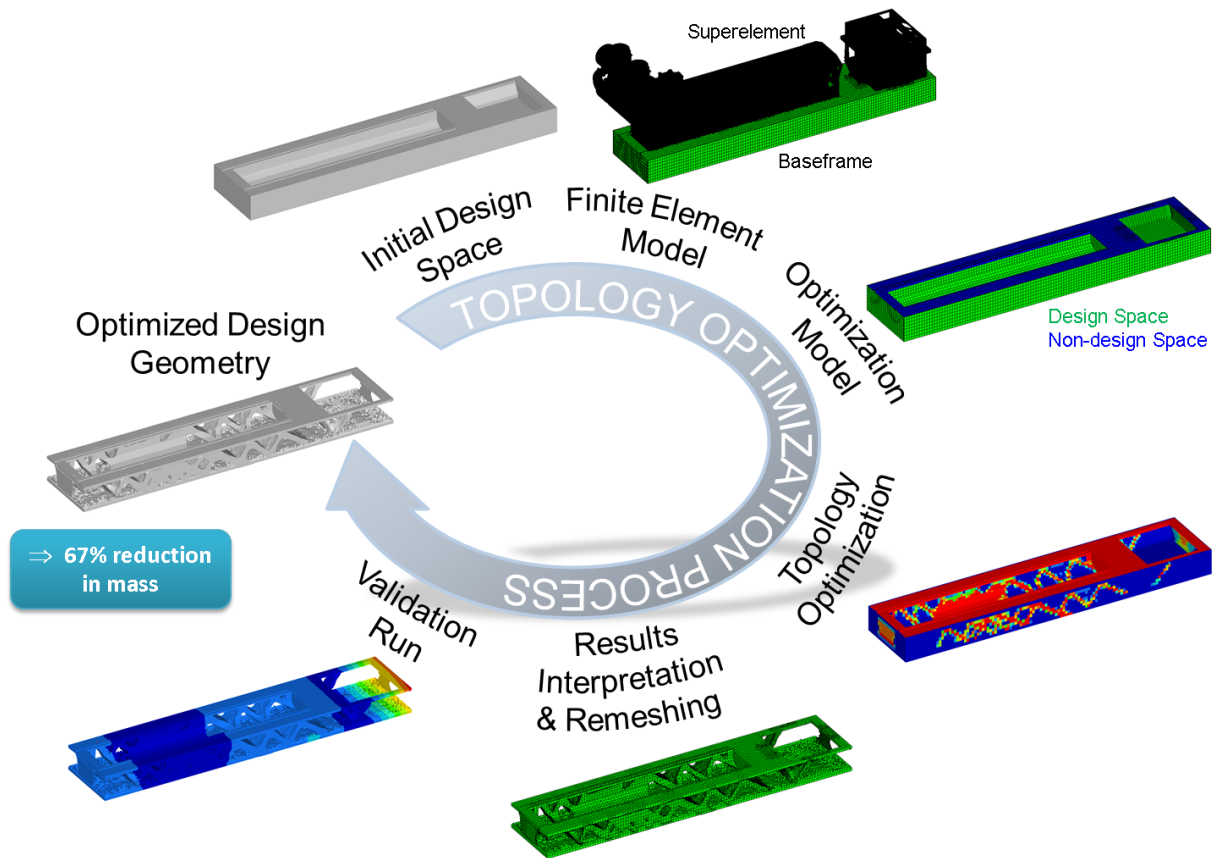


Figure 3. Typical way process might proceed for the base frame topology optimization.

For the analysis, a full FEM model of the engine, generator and base frame was created for use in a modal analysis calculation. As only the base frame would change during the optimization process, superelements were also generated for the engine and generator. After a working model was created, an initial topology optimization was performed. This was followed by the creation of shape variables using HyperMorph, in which the length, width and height of the base frame mesh could be varied. These shape variables were used as the design variables in the DOE routine, which was implemented in a HPC environment. The optimization results were automatically interpreted and a new mesh created, which was then analyzed to see if the design criteria were met. All of these steps are described in detail in the following sections.

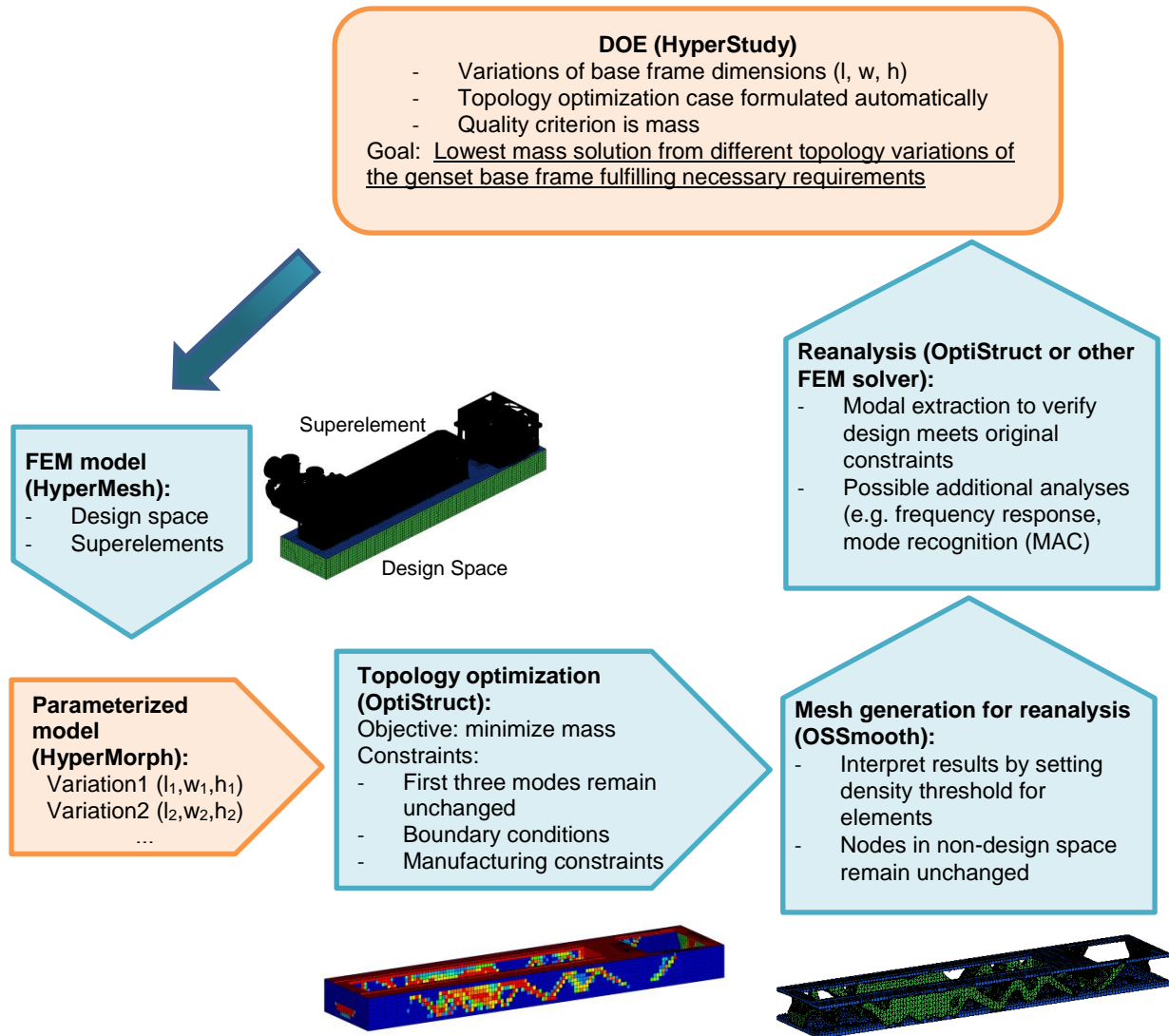


Figure 4. Description of the proposed design of experiments set-up

3.1 Parallelization of OptiStruct

The solver selected for the topology optimization problem is Altair OptiStruct. Parallelization of the software was studied by running a topology optimization of a generic base frame model containing 9450 elements and nearly 130,000 degrees of freedom (DOF). This model was solved utilizing a varying number of CPUs, with the solution time monitored. Knowledge was gained about the efficiency of using multiple processors, and the information was used to select an 'economical' number of CPUs for the chosen analysis.

Table 1 displays the results of the study. Wall time describes the actual time elapsed to get a solution, percent difference gives the change between the given number of CPUs and the level below it, percent reduction describes the difference between the current CPU level and the run that utilizes only 1 CPU, CPU time is the total amount of time spent by the given CPUs, and effective CPU is the ratio between the wall time for the 1 CPU case and the wall time for the given number of CPUs.

Table 1. Results of OptiStruct parallelization study.

CPU's	1	2	4	6	8
Wall time (h)	15.40	10.65	5.95	5.82	5.70
Difference (%)	-	30.84	44.13	2.24	2.01
Reduction, ref. 1 cpu (%)	-	30.84	61.36	62.23	62.99
CPU time (h)	12.92	15.50	12.33	16.27	20.45
Effective CPU	1.00	1.45	2.59	2.65	2.70

Wall time is plotted for varying number of CPUs used for the simulation in Figure 5. It is evident from the results shown in the table and plot that simulation time does not change much when more than 4 CPUs are used. This finding also agrees with tests performed by Altair (the maker of OptiStruct). A further reason to limit the number of CPUs utilized in an OptiStruct simulation is that up to 4 CPUs can be used in a single simulation without an increase in the number of software tokens required. Thus, for the reasons listed, 4 CPUs were chosen for all subsequent OptiStruct simulations.

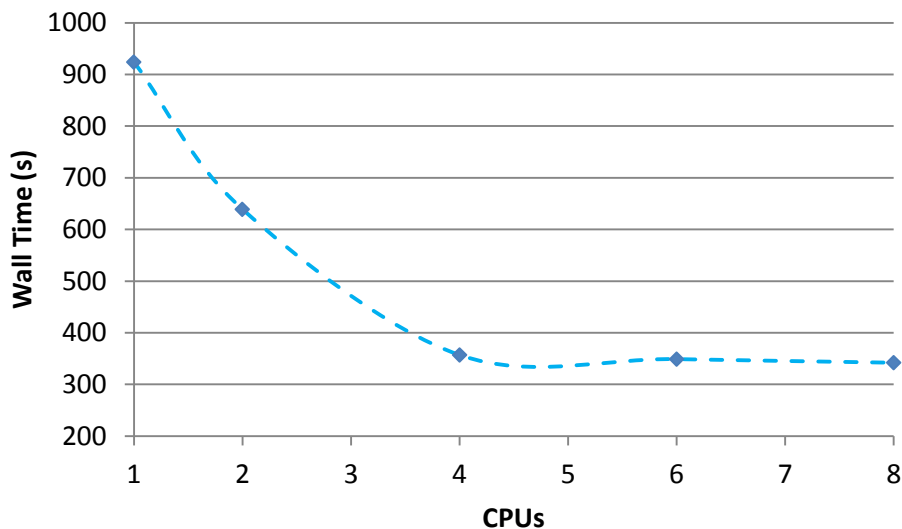


Figure 5. Elapsed simulation time for varying number of CPUs

3.2 FEM model creation

The finite element model was first created in NX Ideas and then exported in Nastran format. It was then imported into HyperMesh, where it was converted to OptiStruct format. The original base frame model was made of shell elements. The base frame was changed into a solid model for the topology optimization, but care was taken so that the nodes attaching the base frame to the engine and generator remained unchanged. The model is displayed in Figure 6.

In order to reduce computation time, superelements were created to represent the engine and generator (Figure 6b). Instructions on how to create the superelements can be found in the OptiStruct User's Guide under the heading "Direct Matrix Input" [2].

The final base frame model consisted of 184k 2nd order elements and 282k nodes (not including the superelements). A structural modal extraction procedure was set up in OptiStruct to compute the normal modes of the base frame up to 50 Hz. When utilizing 4 CPU on the VTT computing cluster, the modal extraction takes 11m 50s.

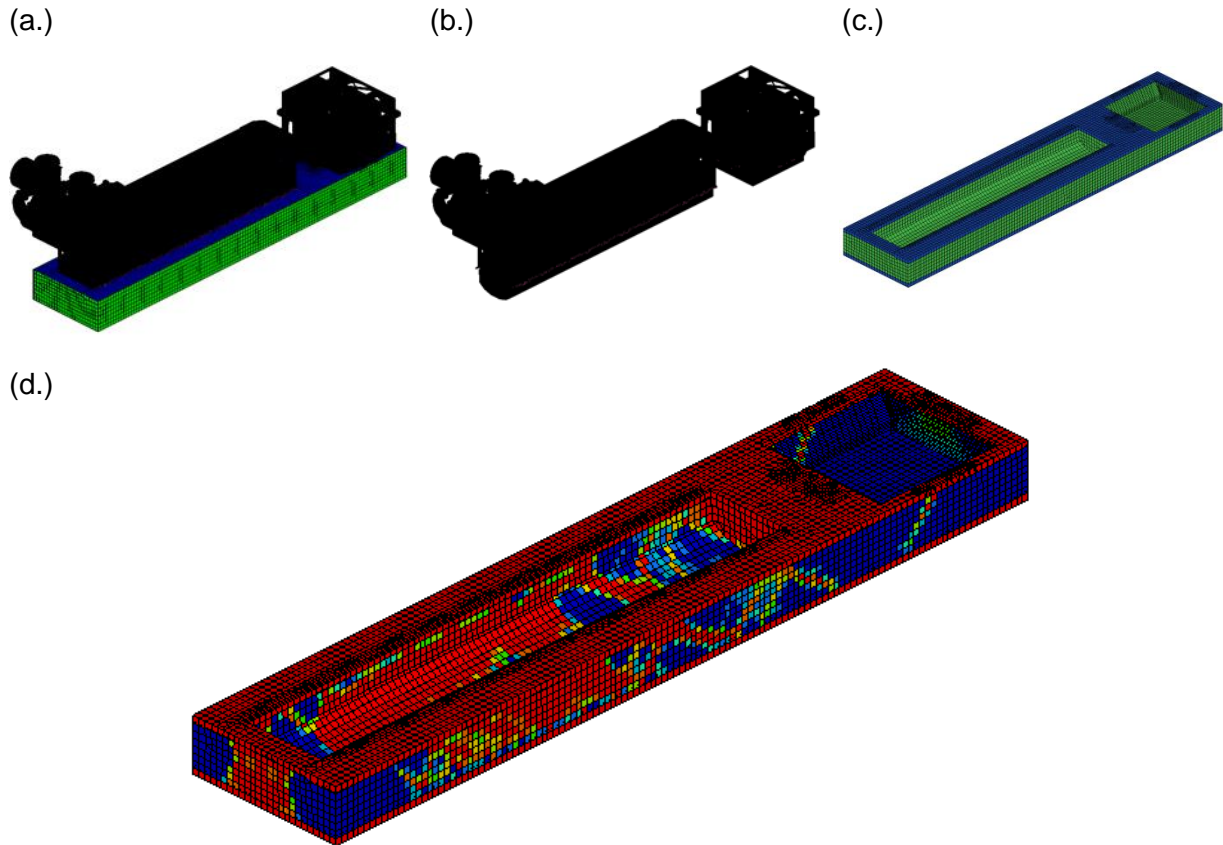


Figure 6. Genset FEM model; (a) engine, generator and base frame, (b) superelements for the engine and generator, (c) base frame model – green elements are allowed to vary during topology optimization while blue elements are fixed, (d) element density plot after topology optimization – red elements have highest density value (1.0) and blue elements have lowest (0.0).

3.3 Topology optimization

Structural optimization can be divided into three subclasses – size, shape, and topology optimization. Topology optimization is the most general of the three, and involves determination of features such as the number, location and shape of holes, and material connectivity [1, 3].

OptiStruct software utilizes the solid isotropic material with penalization (SIMP) method for solving topology optimization problems. With this method, the material density of each element in the model is used directly as a design variable. The density values of the elements vary continuously between 0 and 1, and represent a void or a solid, respectively. Intermediate density values describe a fictitious material, and it is assumed that material stiffness is linearly dependent on density. A penalization strategy is utilized to try to force intermediate density values in the final design toward values of either 1 or 0. Various manufacturing constraints can be applied during the topology optimization (e.g. minimum member size control, draw direction, extrusion, pattern repetition, pattern grouping, etc), which in turn affect the penalty value applied [2]. The general flow of the topology optimization procedure can be found in Figure 7.

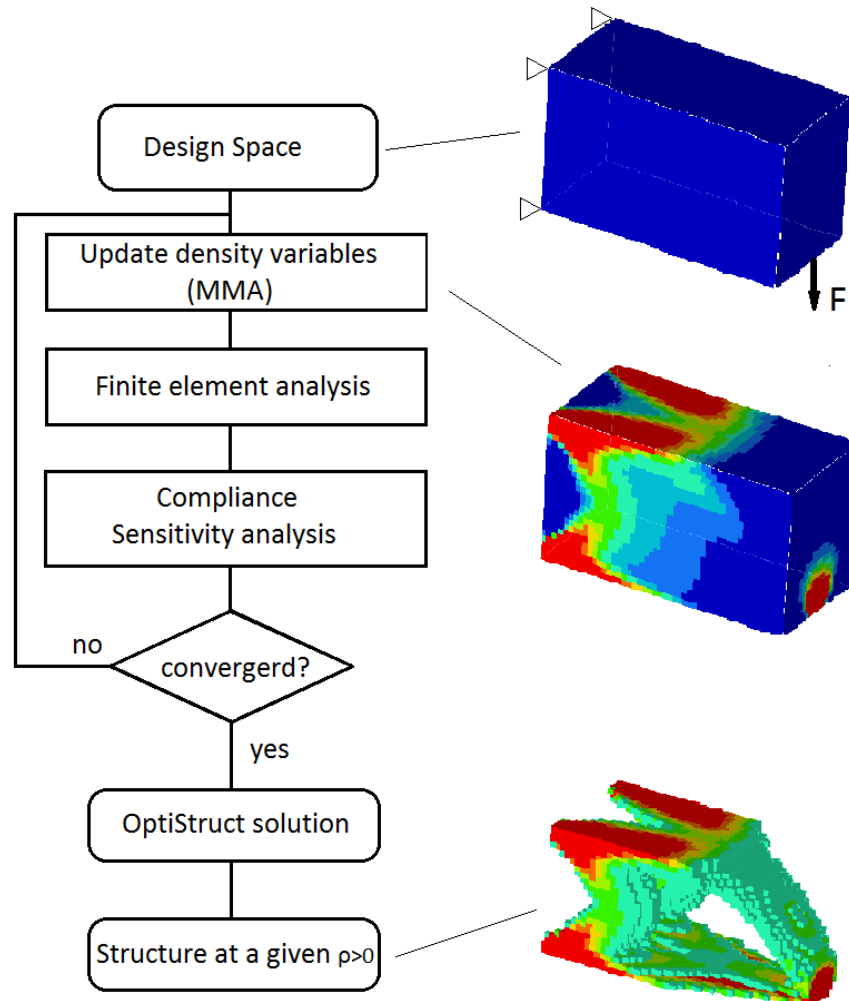


Figure 7. OptiStruct topology optimization iteration scheme [1, 2].

The finite elements of the model are designated as falling within one of two categories: design space, or non-design space. The design space elements are those for which the element density can vary within the topology optimization, and may potentially be removed or altered for the final design. Non-design space elements are those which remain unchanged during the optimization procedure, and are typically the locations where boundary conditions, loads, or other constraints are applied. In the current study, the non-design space (see Figure 6c, blue elements) includes all of the nodes that connect to the superelements as well as part of the lower surface where the steel springs connecting the frame to the concrete floor might be positioned.

The optimization problem in the current study was formulated as follows:

Objective: Minimize mass of the base frame

Constraint: Frequency range of normal modes:

$$f_{1,min} < f_1 < f_{1,max}$$

$$f_{2,min} < f_2 < f_{2,max}$$

$$f_{3,min} < f_3 < f_{3,max}$$

Manufacturing constraints:

- Draw direction
- Pattern grouping (i.e. symmetry plane)

The existing common base frame design works well with the engine and generator set-up, and does not have any natural frequencies that correspond to key engine orders. Thus the first three eigenfrequencies are constrained such that they stay within 1 Hz of the current design. Manufacturing constraints of a single symmetry plane (through the centerline along the length of the base frame) and draw direction (along width of base frame) were implemented.

3.4 Model parameterization

In order to vary the initial design space of the base frame for each run of the DOE, shape variables were created using Altair HyperMorph. This process entails making modification to the mesh by selectively moving groups of nodes, while ensuring that mesh distortion is minimal. This process is referred to as morphing within the Altair user manuals, and this term will be used throughout this text. Morph constraints were used to ensure that nodes in the non-design space would not be moved. The original definition of the base frame design space made it relatively easy to modify the structure's height without moving any restricted nodes. However, as the top surface of the base frame along the entire width and length was considered non-design space, it was not possible to make a uniform size change in these two directions. Thus, a layer of elements was added around the outside of the base frame, as in Figure 8. The outer nodes of these elements were not restricted in any way, and were thus able to move freely to allow changes to be made in the structure width and length.

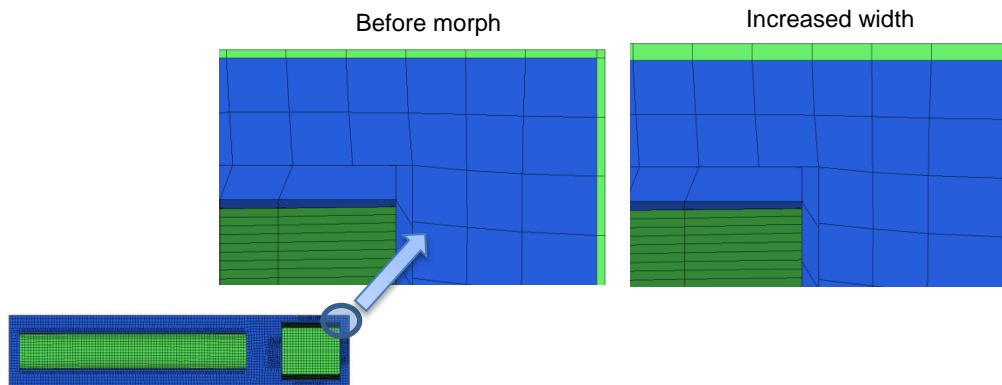


Figure 8. Top view of base frame model with one corner enlarged so that additional element layer (green) around the outside of the model can be seen before and after morphing.

The changes made with HyperMorph to the base frame length, width and height were saved as shape variables for the DOE study. The amount of variation for all three shapes can be parameterized, and thus the magnitude of the length, width and height changes can be scaled automatically within the DOE. Further details on the HyperMorph procedure used for this study are described in Appendix A.

3.5 Design of experiments

A design of experiments was initiated within HyperStudy with the described OptiStruct topology optimization and three base frame shape variables. After indicating the correct FEM model and shape files, HyperStudy conducts a nominal run of the simulation so that responses can be selected for inclusion in the analyses. In this study, the first three eigenfrequencies, percentage decrease in mass, and the number of iterations required to reach a solution to the topology optimization problem were selected.

The shape design variables were defined such that the dimensions of the base frame could change in height from 0-10cm, length 0-3cm, and width 0-3cm. Larger variations in the mesh shape are generally possible, but were difficult to achieve with the configuration of the current model and attached superelements.

A full-factorial DOE was set-up with all 3 shape variables solved at 4 different levels, producing 64 runs. A more detailed description of the procedure followed for creating the DOE can be found in Appendix B.

3.6 Mesh generation for reanalysis after topology optimization

As described previously, the result of the topology optimization is the assignment of mesh densities to each element in order to indicate where material is needed and where it is not (Figure 6d). For elements where the density value is 1, material is necessary; and where the value is 0, material should be removed. However, the result is open to interpretation at all intermediate values. OSSmooth is meant to aid in this design interpretation step, and can facilitate the recovery of a modified geometry for further use in the design process and FEM reanalysis. A flowchart describing how OSSmooth works to interpret OptiStruct optimization results is shown in Figure 9. When interpreting topology optimization results for the purpose of reanalysis, the software can preserve component boundaries for multiple design components, recover geometry with or without an artificial layer of elements around non-design space, tetramesh iso-surfaces by (user-defined) 'property', and preserve boundary conditions upon geometry recovery [2].

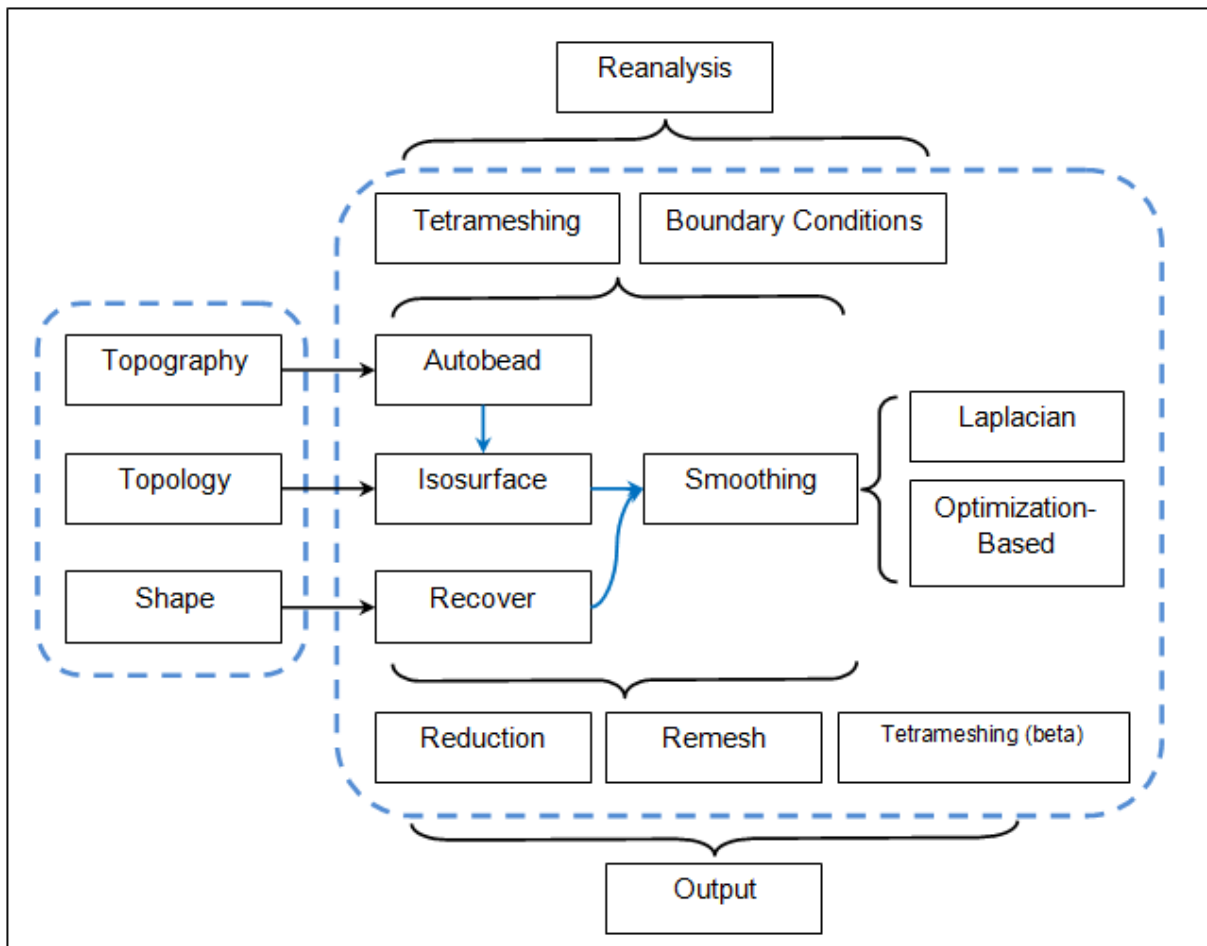


Figure 9. Flowchart describing how OSSmooth interprets OptiStruct optimization results [2]

After the topology optimization was completed in each DOE run, OSSmooth was automatically launched with several element density thresholds, connection detection thresholds, and other interpretation criteria values tested. In cases where the mesh was successfully created, OptiStruct was launched to analyze the new design. The first three eigenfrequencies and the mass of each new design were recorded for comparison with the

original design constraints. Examples of the necessary launch files can be found in Appendix B.

It is also possible to use the DOE to run only the remesh and reanalysis steps. This can prove to be useful if it is found that after the optimization has been performed, the design interpretation provided unsatisfactory results. Rather than having to run the entire simulation again, the results files from the topology optimization can be reused and analyzed with different criteria. In this particular case, this resulted in huge time savings as the topology optimization takes many hours to converge, while the mesh creation and reanalysis can be performed in under 30 minutes.

3.7 Running simulation in HPC environment

The DOE was prepared on the front-end node of a computing cluster, and was launched so that 8 runs ran simultaneously on a single 32 cpu node. Each run used 4 cpu to run the OptiStruct topology optimization. The typical time to achieve a converged solution (per run) was 1.5 days. It is just as easy to launch the simulation to run over numerous computational nodes simultaneously, but doing so requires additional license tokens for the solver. The necessary launch commands and associated text files are also included in Appendix B.

4. Optimization results

The optimization results produced in this study were not subjected to the same level of scrutiny that a real design case might be, as that was beyond the scope of this project. Rather, the main objectives of launching a design of experiments on a computational cluster in order to control a large number of topology optimizations, automatically remesh the resulting density contours subject to various interpretation criteria, and then launch an example reanalysis were accomplished in full. The reanalysis part of this procedure could be expanded to include, for example, frequency response analysis or mode assurance criterion (MAC). The following describes the obtained results.

As mentioned previously, the topology optimization of the base frame was performed for 64 design spaces with varied dimensions. The OptiStruct launch files (*.fem) were automatically generated by the HyperStudy DOE, and a shape file (*.sh) describing the densities of the elements was created during each optimization run. These two files were interpreted in various ways for each run. In the end, there were hundreds of designs to choose from, each having been reanalyzed so that the percent reduction of the structure mass and first three modes were readily available for use in comparison of the resulting designs.

Of the 64 initial topology optimization runs, 16 had to be discounted as the optimization was terminated due to excessive mesh distortion. In every case that failed, at least one of the dimensions of the base frame was at its maximum value. It is thus extremely important to think about the entire process (e.g. initial analysis, shape variable creation, design vs. non-design spaces for topology optimization) when generating the model mesh. The connections to superelements and other boundary conditions affect the possible design space that can be utilized, which in turn dictates how the design space can be altered with shape variables within the DOE set-up. Mesh type and size play a key role in determining whether there will be problems with element distortion and mesh generation for reanalysis after topology optimization.

The 48 topology optimization runs that resulted in a feasible solution were subsequently evaluated using OSSmooth software to interpret the results and generate a mesh of the design for reanalysis. The criteria used during this step for results interpretation can significantly affect the design and the ability of the software to create a usable mesh. As an example, run 36 of the topology optimization DOE is examined, which has base frame height

increase of 5%, width increase of nearly 1% and no length change as compared to the original design space. Figure 10 shows the element density color map resulting from the topology optimization in run 36. The non-design space and all elements having a density value of 1 (solid) are shown in red, while blue represents element densities of 0 (void). Table 2 describes nine different interpretations of the results by OSSmooth, and includes the difference between the target and simulated first three eigenfrequencies and percent mass reduction of the base frame for each variation. The element density threshold values tested varied from 0.1-0.55, the low density threshold for connection detection varied from 0.15-0.25 on those runs in which it was provided, the option to recover draw constraints in the optimization was not used as it did not lead to usable meshes, and in all but one case the boundary was included in the Laplacian smoothing. The eigenfrequencies in the table highlighted in green met the original target of being within 1 Hz of those of the current base frame design, while the eigenfrequencies highlighted in yellow are within 1.5 Hz. Figure 11 shows two of the run 36 mesh configurations produced by OSSmooth. There are only very slight differences in material placement, but the consequence is that one of the design interpretations very nearly met the design criteria (*H*), while the other (*F*) had problems with a hanging or unconnected part (circled in red) on the far right side of the base frame. The connection problem that was shown in the figure happened frequently with some of the interpretation criteria. If this problem could not be solved by simply changing the interpretation values, this could be addressed with some mesh cleanup. But this likely needs to be done on a case by case basis rather than automatically within the simulation scripts. For the current study, simply modifying the OSSmooth interpretation criteria was sufficient.

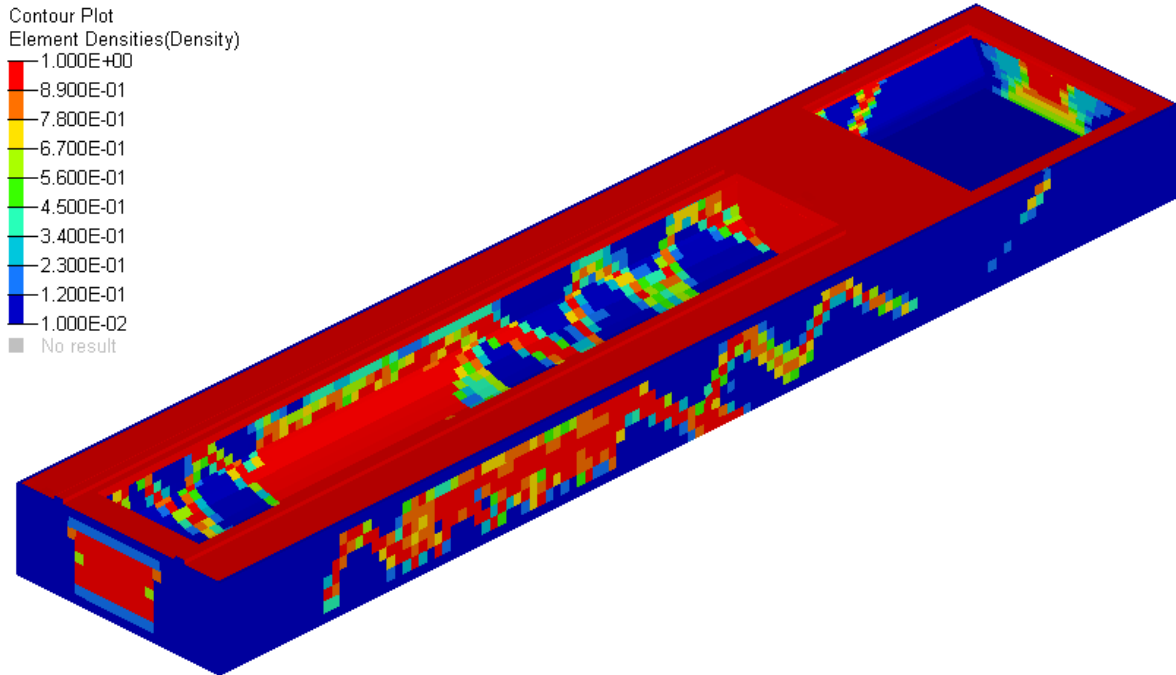


Figure 10. Topology optimization element density plot for DOE run 36.

Table 2. Nine OSSmooth interpretations of the topology optimization result from DOE run 36 and the corresponding reanalysis findings

OSSmooth Set-up					Reanalysis Results for DOE Run 36				
	Density Threshold	Connect Detect, Low Threshold	Draw Recovery	Include Boundary in Laplacian Smoothing	Eigenfrequency (Target-Simulated)			Mass Reduction (%)	Mesh Problems
					Δf_1 (Hz)	Δf_2 (Hz)	Δf_3 (Hz)		
A	0.1	-	No	Yes	1.91	2.87	0.46	36.97	No
B	0.15	-	No	Yes	1.65	2.76	0.44	39.47	No
C	0.2	-	No	Yes	1.32	2.63	0.41	41.64	No
D	0.25	-	No	Yes	-6.54	-5.18	-7.58	45.39	No
E	0.3	-	No	Yes	X	X	X	X	Yes
F	0.35	-	No	Yes	-0.01	-1.97	-4.07	47.71	Yes
G	0.5	0.15	No	No	-0.11	1.51	-0.21	55.28	No
H	0.5	0.2	No	Yes	-0.29	1.18	-0.23	55.46	No
I	0.55	0.25	No	Yes	-0.11	1.51	-0.21	55.28	No

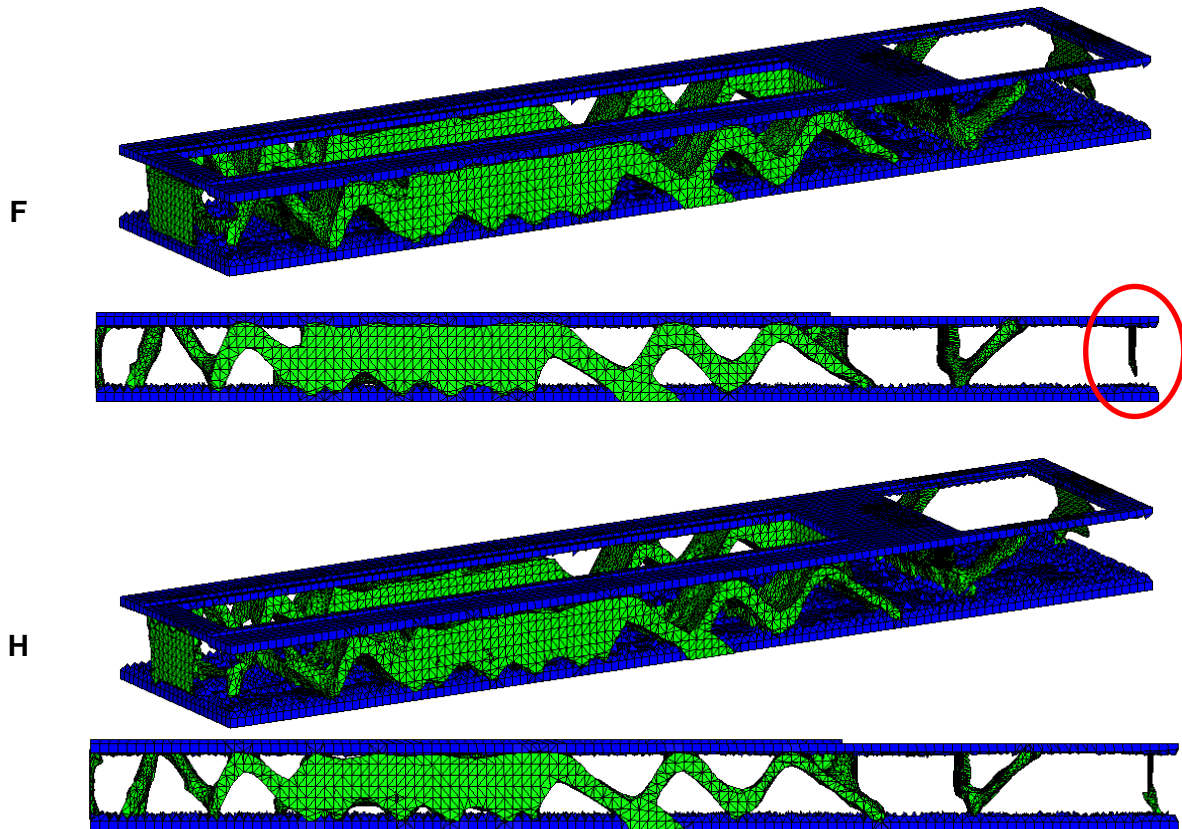


Figure 11. Mesh for reanalysis generated from DOE run 36 topology optimization interpreted by OSSmooth using the criteria for variation *F* and *H* described in Table 2.

The original design criterion for the base frame was that the first three eigenfrequencies remain within 1 Hz of the current design. This proved to be an unattainable target with the optimization set-up used, however, there were a large number of designs for which the first and third eigenfrequencies met the original criterion. Additionally, there were 19 design variations for which the first three eigenfrequencies were within 1.5 Hz of the current design. Ten of these designs were obtained using OSSmooth set-up *H*, while the others came from *G* and *I*. So for the models presented here, a higher threshold value combined with a lower density threshold for connectivity detection produced the best interpretations. There were also 4 runs of the DOE that could be interpreted by set-up *G*, *H*, or *I* with good results (i.e. eigenfrequencies within 1.5 Hz), and these designs can be found in Figure 12.

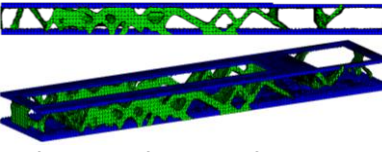
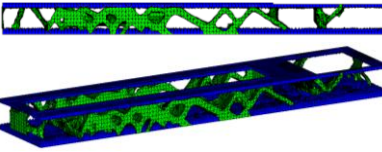
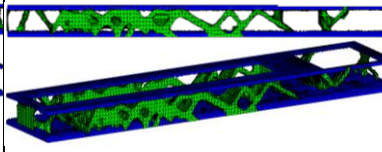
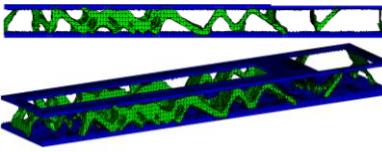
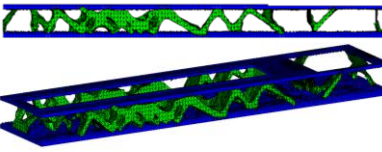
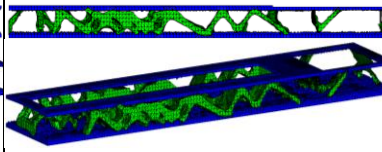
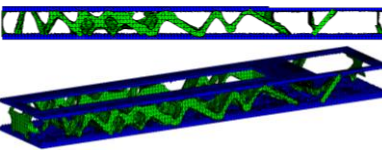
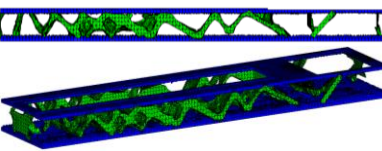
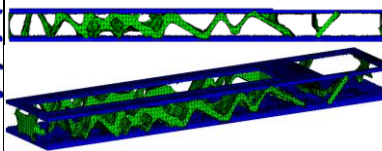
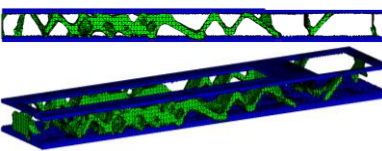
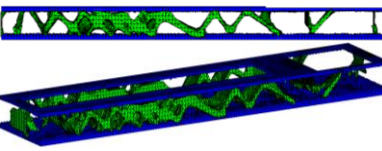
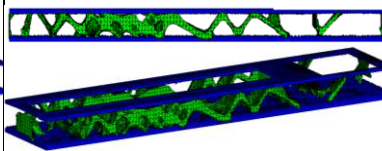
	G	H	I
Run 29	 $\Delta f_1=-0.1, \Delta f_2=-1.5, \Delta f_3=-0.4$ Hz	 $\Delta f_1=-0.1, \Delta f_2=1.5, \Delta f_3=-0.4$ Hz	 $\Delta f_1=-0.1, \Delta f_2=1.5, \Delta f_3=-0.4$ Hz
Run 50	 $\Delta f_1=-0.6, \Delta f_2=1.2, \Delta f_3=-0.5$ Hz	 $\Delta f_1=-0.6, \Delta f_2=1.3, \Delta f_3=-0.4$ Hz	 $\Delta f_1=-0.6, \Delta f_2=1.2, \Delta f_3=-0.5$ Hz
Run 55	 $\Delta f_1=-0.5, \Delta f_2=1.3, \Delta f_3=-0.7$ Hz	 $\Delta f_1=-0.5, \Delta f_2=1.3, \Delta f_3=-0.7$ Hz	 $\Delta f_1=-0.5, \Delta f_2=1.3, \Delta f_3=0.7$ Hz
Run 58	 $\Delta f_1=-0.5, \Delta f_2=1.4, \Delta f_3=-0.6$ Hz	 $\Delta f_1=-0.6, \Delta f_2=1.4, \Delta f_3=-0.6$ Hz	 $\Delta f_1=0.5, \Delta f_2=1.4, \Delta f_3=0.6$ Hz

Figure 12. Topology optimization designs for 4 different initial design spaces (i.e. DOE runs) and 3 variations of the OSSmooth interpretation criteria. The difference between the target and simulated first 3 eigenfrequencies are shown for each case.

5. Limitations

The current study encountered a few limitations which will be summarized here. First, it was only possible to produce relatively small changes in the design space due to restrictions produced by the mesh, attachment to superelements, and other imposed boundary conditions. Also, the size of the model involved in the case study made making changes to any of the base frame nodes attached to the superelements a very time-consuming proposition. A further consequence of the model size was that it took over a day to run a single topology optimization. This affected the number of changes that could be made to the initial model set-up during the span of the project. A final limitation was due to the software chosen for the study. As it was not open-source code, only a limited number of tokens were available for the study. There were considerably more CPUs available for the study on the computing cluster, which in turn would have reduced the total time necessary to run the DOE.

6. Conclusions

This study sought to investigate the use of a HPC environment to efficiently automate some of the early steps in the design process. This was achieved by creating a finite element model of the chosen case study (genset base frame), creating superelements for large, complex components that were attached (engine and generator), preparing a topology

optimization model, generating shape variables that altered the length, width and height of the design space, and launching a design of experiments on a computing cluster so that 64 variations of the topology optimization were run. Upon completion of each optimization run, the DOE automatically launched a program to interpret the results in several different ways, and then subsequently submitted the new design for reanalysis based on the initial design criteria. The results of the reanalysis were automatically tabulated for easy comparison of designs. The DOE set-up was also flexible enough to allow for additional topology optimization results interpretation and reanalysis after the initial runs were completed. The final result was hundreds of designs from which it was easy to select those which were most able to achieve the design goals.

There are two points within this process in which the number of potential designs increases. The first is due to the change in design variable of the DOE (in this case initial design space), and the second is when the topology optimization is interpreted using OSSmooth. There is no requirement that shape variables be used as the design variables in the DOE, it would be just as easy to vary material properties, for example. It is also possible to use this type of analysis set-up to only focus on the design interpretation step. In that case, the design variables of the DOE could be the various interpretation parameters used (e.g. element density threshold, minimum element density threshold for connection detection, etc.). Again the result would be a large number of working FEM meshes that have been reanalyzed based on initial design criteria, with results tabulated in a convenient fashion.

7. Suggestions for future work

A possible next step for this work would be to analyze a smaller topology optimization problem in which there are fewer constraints on the shape of the design. This would allow the capabilities of the software and this method to really be explored and tested.

Another interesting topic could be investigating the use of this type of topology scheme for creation of 3D printed parts. In these cases, traditional manufacturing constraints do not need to be considered, however those particular to the additive manufacturing process would need to be considered and implemented. Additionally, material property definitions within the models should be examined.

A further potential topic for study could be topology optimization of shell structures. The common base frame used for the genset discussed in this study is in reality a thin-walled structure with a number of supporting ribs or panels. Topology optimization of multi-layer shell structures can be used to help identify the locations for optimal rib placement.

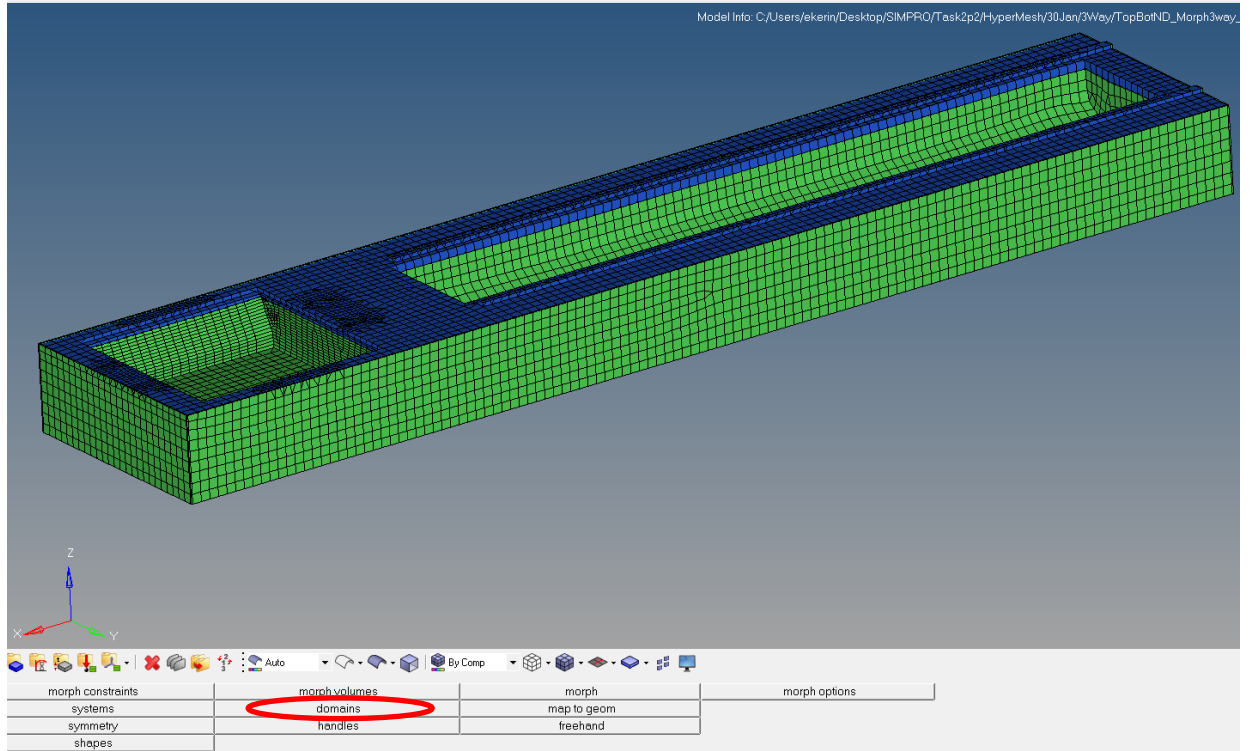
References

- [1] M. Bendsøe and O. Sigmund, *Topology Optimization: Theory, Methods and Applications*, Berlin: Springer, 2003.
 - [2] Altair Engineering, Inc., "HyperWorks 13.0 Online Help," 2015.
 - [3] J. Hämäläinen, "Substructure topology optimization of an electric machine," Aalto University, Espoo, 2013.
-

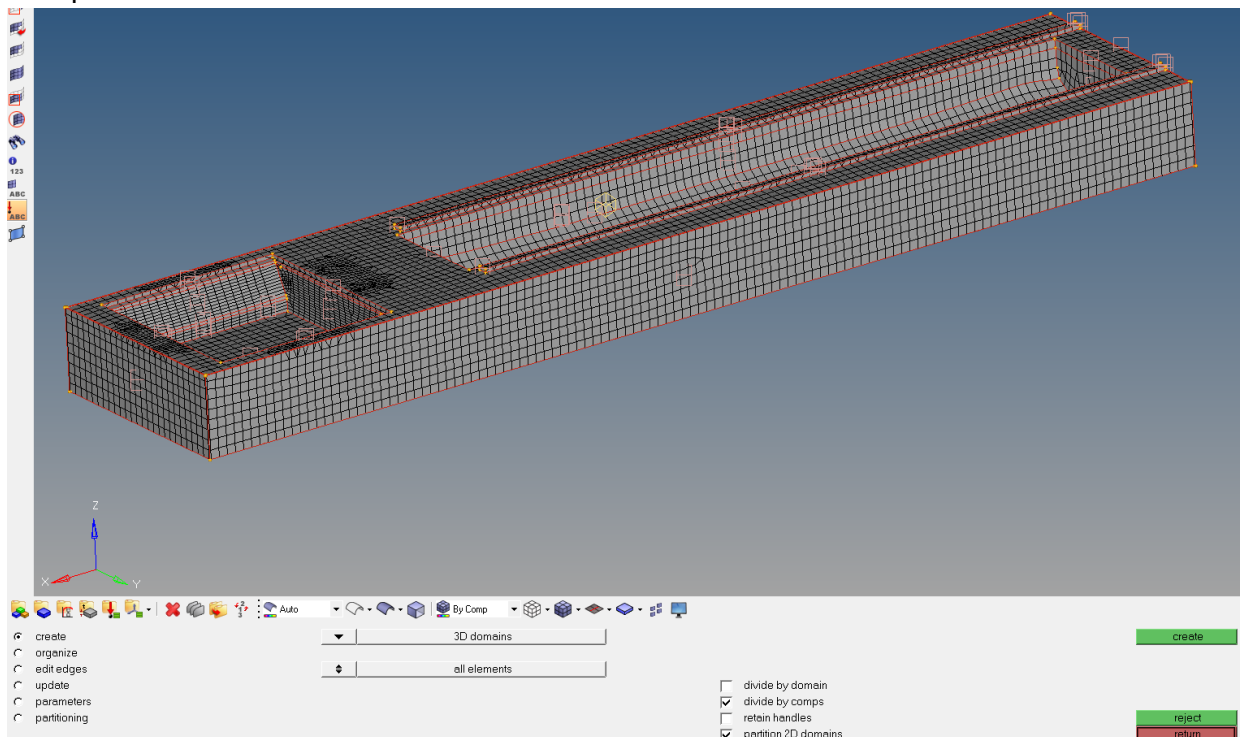
Appendix A – Creation of shape variables with HyperMorph

The following describes how to use HyperMorph to create and export a shape variable for use in a HyperStudy DOE study.

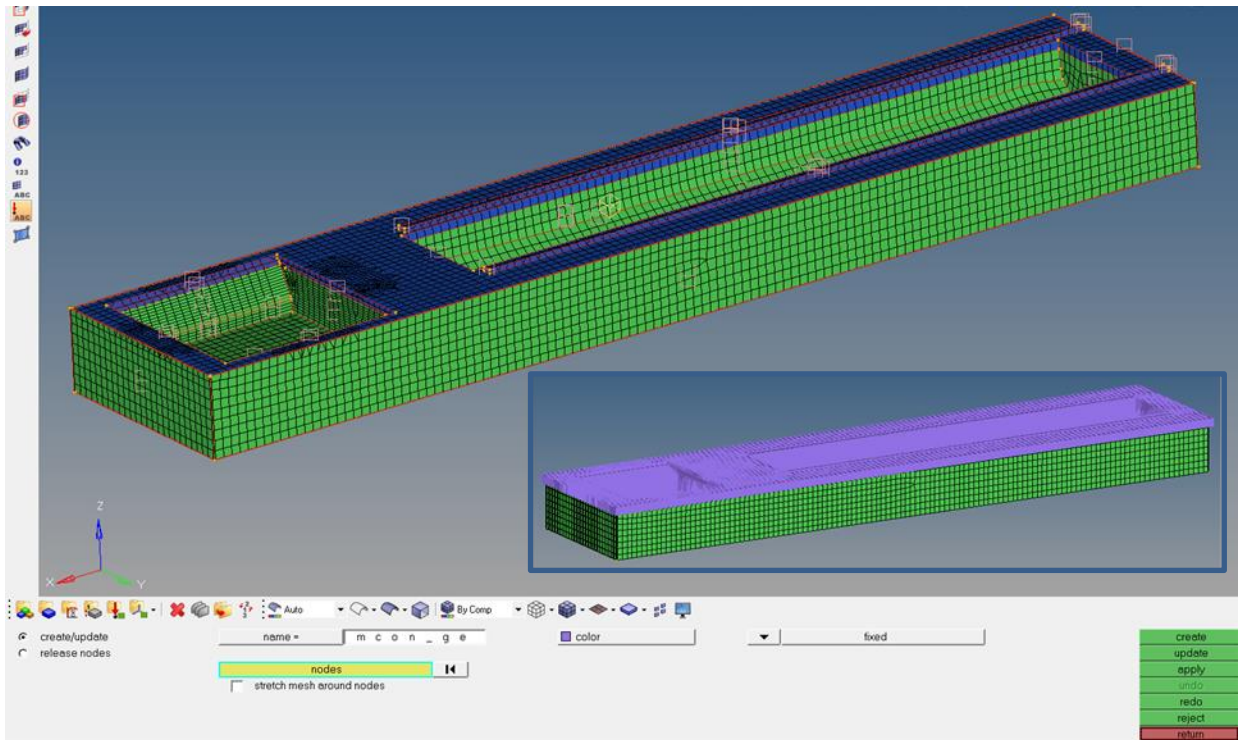
1. Go to Tools->HyperMorph -> Domains



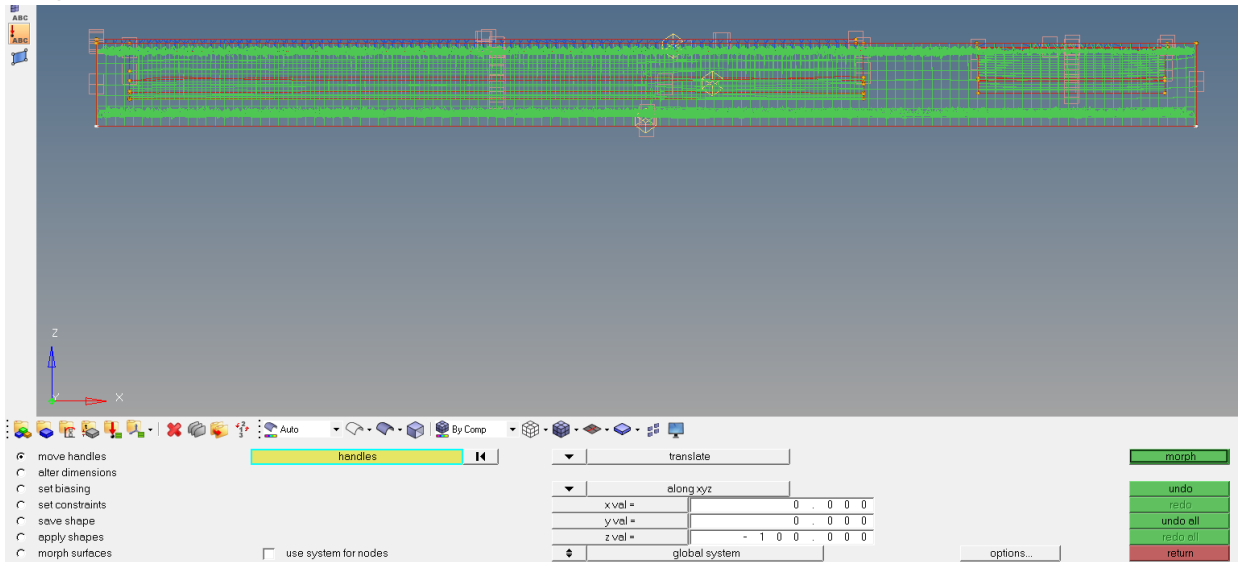
2. Create 3D domains: select 'all elements' and check the boxes by 'divide by comps' and 'partition 2D domains' and click 'create'



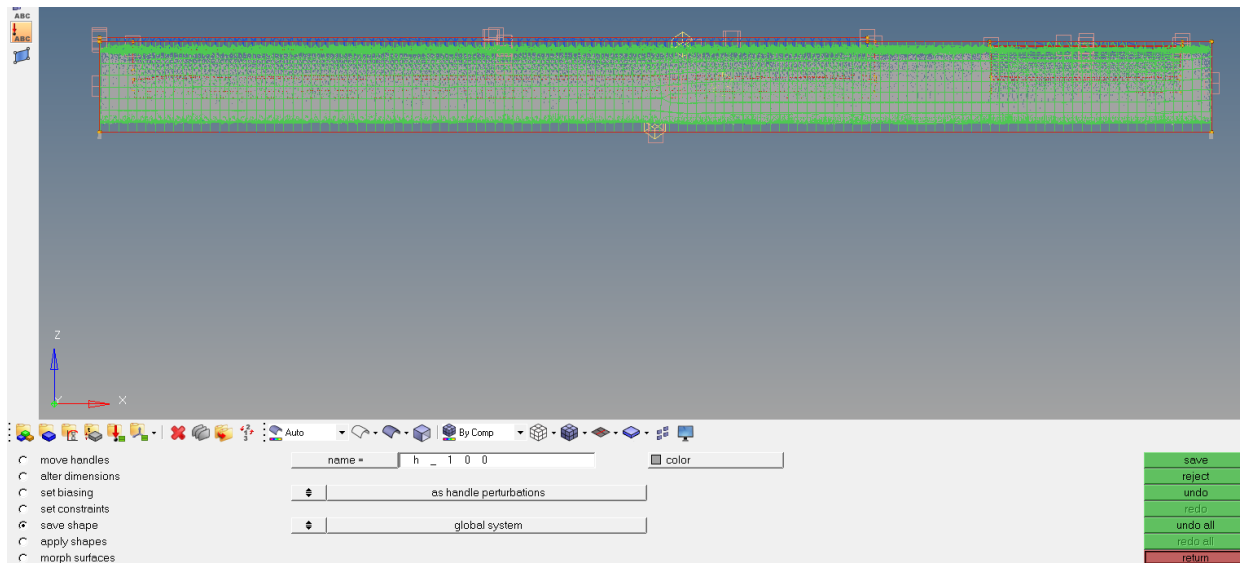
3. Select the nodes to constrain during morphing (e.g. nodes attached to superelements): HyperMorph->Morph Constraints. (The selected nodes in this case are shown in purple in the inset figure.)



4. Morph handles: Tools->HyperMorph->Morph, select radio button 'move handles', use arrow selection to select 'translate', and choose the handles corresponding to a given morph direction



5. Save shape: Tools->HyperMorph->Morph, select radio button 'save shape', give name and click 'save' button



6. Undo morph: click 'undo all' button to go back to original state
7. Create design variables for the saved shapes: Analysis->Optimization->shape, make sure the 'desvar' and 'create' radio buttons are selected, if more than one shape has been made click the down arrow and select 'multiple desvars', choose initial value, lower bound and upper bound, and click 'create' button.
8. Export shape information: Analysis->Optimization->shape, select the 'export' radio button, set analysis code = HyperStudy and sub-code = OptiStruct, and click 'export as...' to give the saved files a name and directory. This process should result in the creation of two files, having the extensions .shp and .optistruct.node.tpl. Both files are necessary for creation of the HyperStudy DOE study.

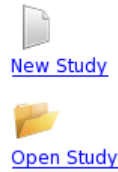
Appendix B – Design of experiments set-up in HyperStudy and launch on computational cluster

The following is a description of how to set-up and run a HyperStudy DOE on the front-end node of a computing cluster, and how to launch the various analyses on computation nodes. The cluster described uses the Sun Grid Engine (SGE) queuing system, and thus the launch commands given are meant for this system. Examples of the actual launch files are shown at the end of the set-up description.

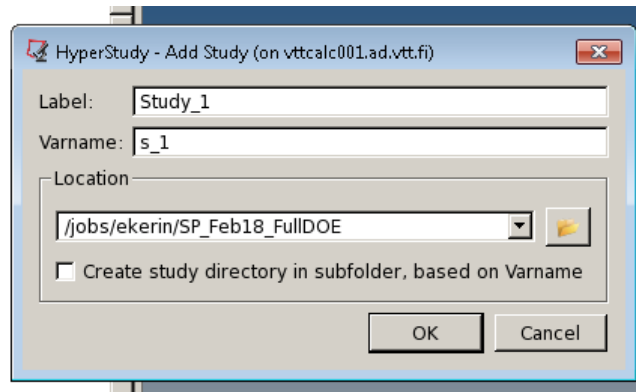
1. Create a directory where study will be run and make it current
2. Open HS on front end node with this command:
`.../hyperworks/12.0/altair/scripts/hst_ng`

3. Create new study

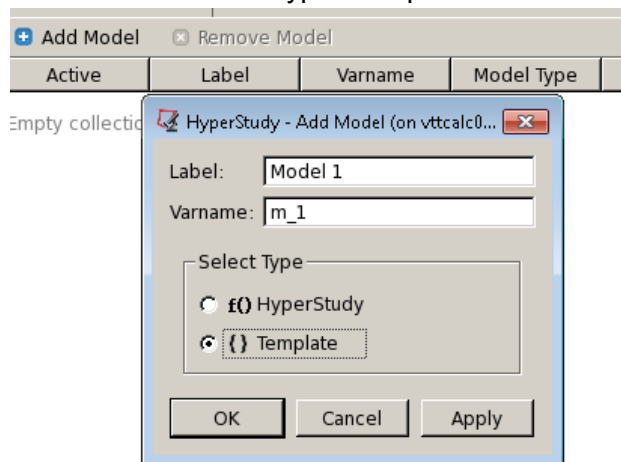
Start a Study:



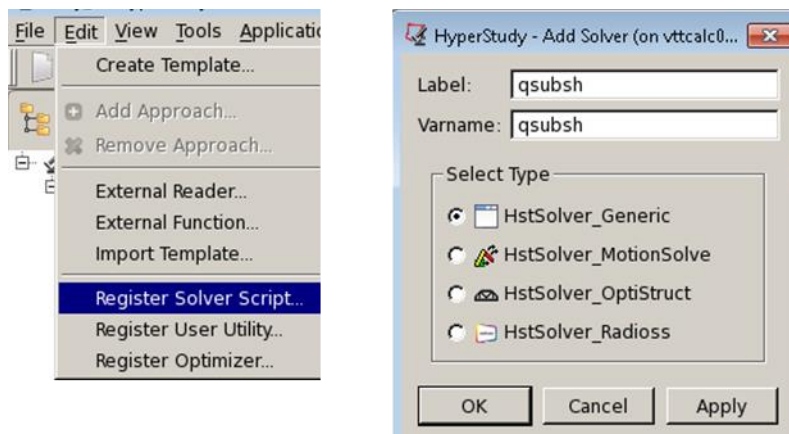
Recent Studies:



4. Add a new model of type 'Template'



5. Register solver script – give path name for the shell file you have created with the command to submit the job to the cluster queue



+ Add Solver Script - Remove Solver Script			
	Label	Varname	Path
1	RADIOSS	radioss	/share/apps/hyperworks/12.0/altair/scripts/radioss
2	OptiStruct	os	/share/apps/hyperworks/12.0/altair/scripts/optistruct
3	Templex	templex	/share/apps/hyperworks/12.0/altair/scripts/templex
4	HyperXtrude	hx	/share/apps/hyperworks/12.0/altair/scripts/hx
5	MotionSolve - standalone	ms	/share/apps/hyperworks/12.0/altair/scripts/motionsolve
6	Python	py	/share/apps/hyperworks/12.0/altair/hw/python/python27/linux64/python
7	TCL	tcl	/share/apps/hyperworks/12.0/altair/scripts/hw_tclsh
8	HyperMath	hmath	/share/apps/hyperworks/12.0/altair/scripts/hmathbatch
9	qsubsh	qsubsh	/jobs/ekerin/SP_Feb18_FullDOE/qsub.sh

6. Define model:

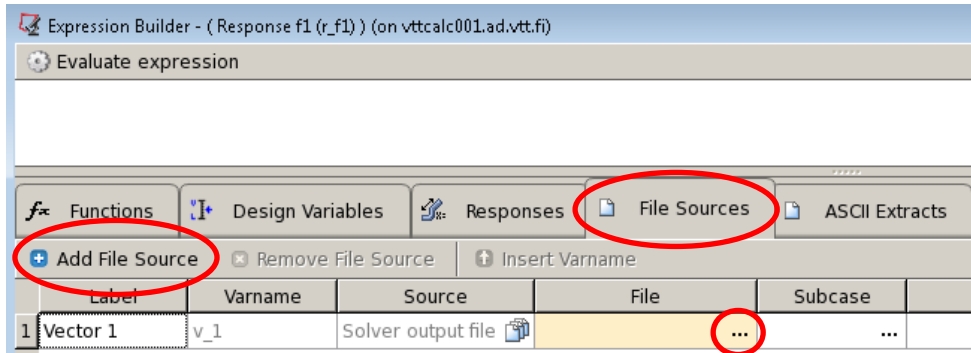
- a. Resource: template (.tpl) file that has been created that introduces shape design variables (from .optistruct.node.tpl file) to the OptiStruct launch file (.fem)
- b. Solver input file: several files need to be listed, including the OptiStruct model file (.fem) and also the text files launching the OptiStruct and OSSmooth jobs (e.g. submit_optistruct1.txt, submit_optistruct2.txt, submit_optistruct3.txt) and the OSSmooth (.tcl) files called by submit_optistruct2.txt (OSS_script1.tcl, OSS_script2.tcl, OSS_script3.tcl)
- c. Solver execution script: choose the script that you have registered for this job that includes qsub command for submitting the OptiStruct jobs in the cluster queue (e.g. qsub.sh)
- d. Solver input arguments: leave blank, as these arguments are given in the solver input text file

Label	Varname	Model Type	Resource	Solver input file	Solver execution script
Model 1	m_1	Template	{ } TopBotND_Morph3way_30Jan.tpl	TopBotND_Morph3way_30Jan.fem;submit_optistruct.txt	qsubsh (qsubsh)

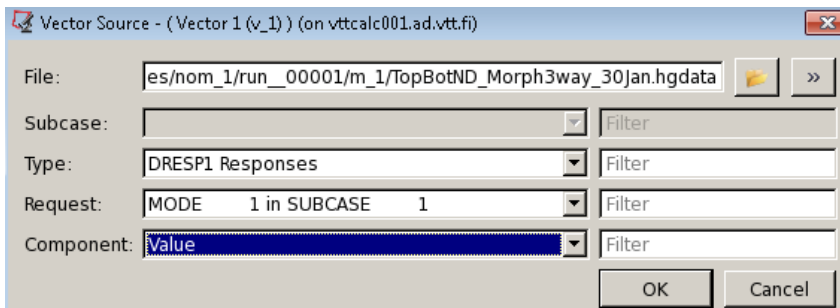
7. Click 'Import Variables'
8. Click 'Apply' to generate design for nominal run
9. Click 'Evaluate Tasks' in order to start nominal run. Make sure that 'Write', 'Execute', and 'Extract' boxes are all checked.
10. After the nominal run has finished, click 'Add Response' to create the responses that will be monitored during the course of the DOE. For this example, the first three normal modes, the percent change in base frame mass, and the number of iterations needed to complete the topology optimization will be monitored. All of this information can be found in the .hgdata file that is automatically created during the optimization. In order to create the response, click on '...' in the Expression column so that the Expression Builder page pops up.

Define responses						
+ Add Response - Remove Response						
Active	Add	Label	Varname	Expression	Value	Comment
<input checked="" type="checkbox"/>		Response f1	r_f1	...	NotExtracted	...

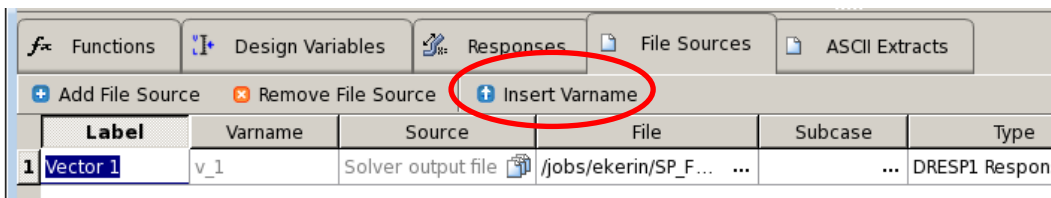
Next click on the 'File Sources' tab and then click 'Add File Source'. Click on the '...' in the File column in order to choose the results file.



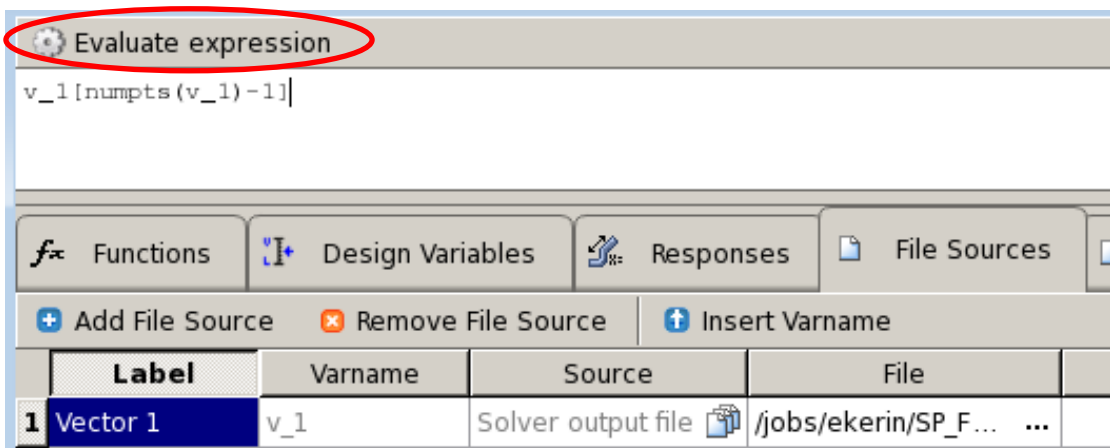
Select the m_1 folder to see the results from the nominal run, and then click on the file with the .hgdata extension. In order to get mass or frequency information, select the type DRESP1 Responses and the Request as either TOTAL MASS or MODE. In order to see the iteration numbers, choose type Iteration.



With the newly created results vector highlighted, click 'Insert Varname' so that the result is added to the Expression Builder.



If the first result is needed (i.e. from iteration 1), then the vector index should be set to 0. If the final result is necessary, use the numpts function as shown below. Click 'Evaluate expression' to check that the response value is as expected.



Define responses				
+ Add Response - Remove Response				
	Active	Label	Varname	Expression
1	<input checked="" type="checkbox"/>	Response f1	r_f1	v_1[numpts(v_1)-1] ...
2	<input checked="" type="checkbox"/>	Response f2	r_f2	v_2[numpts(v_2)-1] ...
3	<input checked="" type="checkbox"/>	Response f3	r_f3	v_3[numpts(v_3)-1] ...
4	<input checked="" type="checkbox"/>	Response 4massc...	r_4masschan...	(v_4[0]-v_4[numpts(v_4)-1])*100/... ...
5	<input checked="" type="checkbox"/>	Response 5iter	r_5iter	v_5[numpts(v_5)-1] ...

11. Click 'Next' until the option Add Approach appears. Select DOE.

HyperStudy - Add Approach (on vttc...)

Label:

Varname:

Select Type

- Doe
- Fit
- Optimization
- Stochastic

OK Cancel Apply

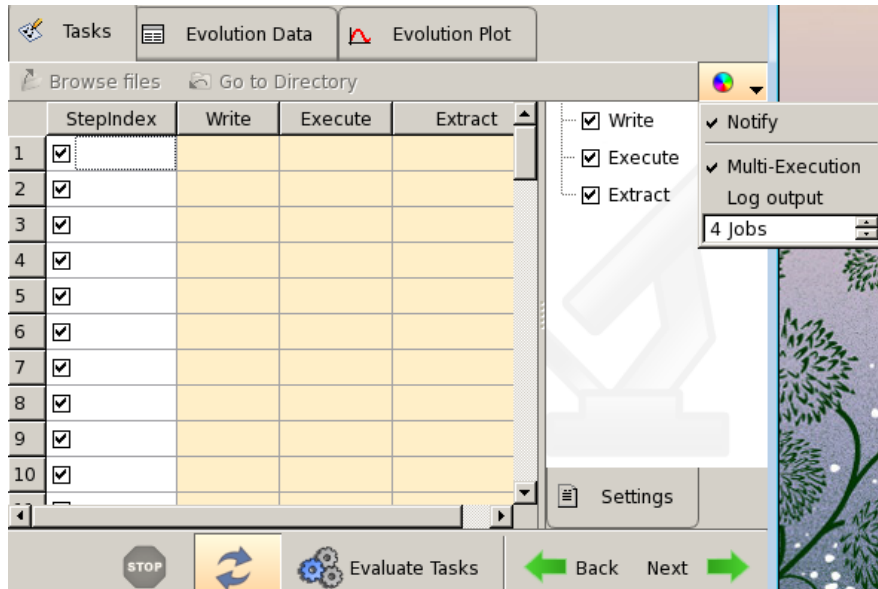
12. If you are happy with the design variables and responses that you have created, click 'Next' until the Specifications tab appears. Here you choose the type of DOE (e.g. Full Factorial), the levels for each design variable (e.g. 4, 4, 4), see the total number of runs (e.g. 64) and click 'Apply'.

Specifications					Uncontrolled Specifications		
	Mode	Label	Varname	Details	Label	Varname	Levels
1	<input checked="" type="radio"/>	Full Factorial	FullFact		1 h_100	m_1_h_100	4
2	<input type="radio"/>	Fractional Factorial	FracFact		2 l_40	m_1_l_40	4
3	<input type="radio"/>	Central Composite	Ccd		3 w_40	m_1_w_40	4
4	<input type="radio"/>	Latin HyperCube	LatinHyperCu...				
5	<input type="radio"/>	Hammersley	Hammersley				
6	<input type="radio"/>	Plackett Burman	PlackBurm				
7	<input type="radio"/>	Box Behnken	Box				
8	<input type="radio"/>	User Defined	User				
9	<input type="radio"/>	Run Matrix	RunMatrix				
10	<input type="radio"/>	None	None				

Parameter Levels Controlled int

Apply ← B

13. Before launching the DOE, the number of jobs to be run simultaneously should be selected. This will obviously affect the total time for the study to be completed, and also the number of HyperWorks tokens that will be in use. Additionally, if you want to launch only a few runs, you can deselect runs by clicking on the check mark next to the run number. Click 'Evaluate Tasks' to launch the DOE.



Example Files Needed in DOE

Shell File (qsub.sh)

```
#!/bin/bash
qsub -sync y -q all.q@compute-1-0 submit_optistruct1.txt
qsub -sync y -q all.q@compute-1-0 submit_optistruct2.txt
qsub -sync y -q all.q@compute-1-0 submit_optistruct3.txt
```

OptiStruct Launch File – Original Analysis (submit_optistruct1.txt)

```
#!/bin/sh
# Used command-line interpreter
#$ -S /bin/bash
# The name of the job
#$ -N TopBotND_Morph3way_30Jan.fem
# Use this folder as a working folder
#$ -cwd
# Write one output file and give its name
#$ -j y
#$ -o PART1_TopBotND_Morph3way_30Jan.out
# Set the OpenMPI parallel environment and the number of processors
#$ -pe orte 4
# Pass the environmental variables
#$ -V
# Reserve resources for the parallel job
#$ -R y
# Optistruct specific command
/share/apps/hyperworks/12.0/altair/scripts/optistruct TopBotND_Morph3way_30Jan.fem -out
-cpu 4 -licwait
```

OSSmooth Launch File (submit_optistruct2.txt)

```
#!/bin/sh
# Used command-line interpreter
#$ -S /bin/bash
# The name of the job
#$ -N PART2_TopBotND_Morph3way_30Jan.fem
# Use this folder as a working folder
#$ -cwd
# Write one output file and give its name
#$ -j y
#$ -o PART2_TopBotND_Morph3way_30Jan.out
# Set the OpenMPI parallel environment and the number of processors
#$ -pe orte 1
# Pass the environmental variables
#$ -V
# Reserve resources for the parallel job
#$ -R y
# Optistruct specific command
(/share/apps/hyperworks/12.0/altair/scripts/hmbatch -tcl
$STUDY_RUN_PATH/OSS_script1.tcl $1 -nobg) & sleep 60 ; kill $!
(/share/apps/hyperworks/12.0/altair/scripts/hmbatch -tcl
$STUDY_RUN_PATH/OSS_script2.tcl $1 -nobg) & sleep 60 ; kill $!
(/share/apps/hyperworks/12.0/altair/scripts/hmbatch -tcl
$STUDY_RUN_PATH/OSS_script3.tcl $1 -nobg) & sleep 60 ; kill $!
```

OptiStruct Launch File – Reanalysis of Optimized Design (submit_optistruct3.txt)

```
#!/bin/sh
# Used command-line interpreter
#$ -S /bin/bash
# The name of the job
#$ -N PART3_TopBotND_Morph3way_30Jan.fem
# Use this folder as a working folder
#$ -cwd
# Write one output file and give its name
#$ -j y
#$ -o PART3_TopBotND_Morph3way_30Jan.out
# Set the OpenMPI parallel environment and the number of processors
#$ -pe orte 4
# Pass the environmental variables
#$ -V
# Reserve resources for the parallel job
#$ -R y
# Optistruct specific command
/share/apps/hyperworks/12.0/altair/scripts/optistruct TopBotND_30Jan_Reanalysis03.fem -
out -cpu 4 -optskip -licwait
/share/apps/hyperworks/12.0/altair/scripts/optistruct TopBotND_30Jan_Reanalysis02.fem -
out -cpu 4 -optskip -licwait
/share/apps/hyperworks/12.0/altair/scripts/optistruct TopBotND_30Jan_Reanalysis01.fem -
out -cpu 4 -optskip -licwait
```

Files Called by OSSmooth Launch File**OSS_script1.tcl**

```
set arg2 [lindex $argv end]
set fe_file [string range $arg2 0 [string length $arg2]-16]TopBotND_Morph3way_30Jan.fem
set sh_file [string range $arg2 0 [string length $arg2]-16]TopBotND_Morph3way_30Jan.sh

set ra01_file [string range $arg2 0 [string length $arg2]-
16]TopBotND_30Jan_Reanalysis01.fem

*createstringarray 1 "isosurf: 3 3 0.1 1 0.1 0 0 10 30 0 0"
*ossmooth_12 0 1 2 1 "$fe_file" "$sh_file" "" 1 0 1 1
set template [hm_info -appinfo SPECIFIEDPATH TEMPLATES_DIR]
*feoutputwithdata "$template/feoutput/optistruct/optistruct" "$ra01_file" 0 0 1 1 0
```

OSS_script2.tcl

```
set arg2 [lindex $argv end]
set fe_file [string range $arg2 0 [string length $arg2]-16]TopBotND_Morph3way_30Jan.fem
set sh_file [string range $arg2 0 [string length $arg2]-16]TopBotND_Morph3way_30Jan.sh

set ra02_file [string range $arg2 0 [string length $arg2]-
16]TopBotND_30Jan_Reanalysis02.fem

*createstringarray 1 "isosurf: 3 3 0.2 1 0.1 0 0 10 30 0 0"
*ossmooth_12 0 1 2 1 "$fe_file" "$sh_file" "" 1 0 1 1
set template [hm_info -appinfo SPECIFIEDPATH TEMPLATES_DIR]
*feoutputwithdata "$template/feoutput/optistruct/optistruct" "$ra02_file" 0 0 1 1 0
```

OSS_script3.tcl

```
set arg2 [lindex $argv end]
set fe_file [string range $arg2 0 [string length $arg2]-16]TopBotND_Morph3way_30Jan.fem
set sh_file [string range $arg2 0 [string length $arg2]-16]TopBotND_Morph3way_30Jan.sh

set ra03_file [string range $arg2 0 [string length $arg2]-
16]TopBotND_30Jan_Reanalysis03.fem

*createstringarray 1 "isosurf: 3 3 0.3 1 0.1 0 0 10 30 0 0"
*ossmooth_12 0 1 2 1 "$fe_file" "$sh_file" "" 1 0 1 1
set template [hm_info -appinfo SPECIFIEDPATH TEMPLATES_DIR]
*feoutputwithdata "$template/feoutput/optistruct/optistruct" "$ra03_file" 0 0 1 1 0
```