Chapter 10
Software Agnostic Hurty/Craig-Bampton Dynamic Model Reduction

Matt Griebel and Taylor Wender

Abstract Launch vehicles (LV), spacecraft (SC), and payload design partners often require a dynamic representation of components to perform coupled loads analysis (CLA). The most common form is a Hurty/Craig-Bampton (HCB) model reduction that provides a mass and stiffness matrix that is representative of the dynamic performance of the component. While this is a routine procedure if the component finite element model (FEM) was constructed in Nastran, the process is not as straight forward if the FEM is in a different software (e.g., Ansys, Abaqus), particularly if the receiving entity (i.e., LV, SC, etc.) requires the HCB in the standard Nastran OP4 format. This paper outlines the process of delivering an HCB in Nastran OP4 format without the need of converting the FEM to Nastran, which can be time consuming and error prone. The resulting HCBs show consistent and accurate results, providing a robust framework for delivering an HCB in Nastran OP4 format regardless of the software the original FEM was constructed in and mostly without the need for any custom scripts or procedures. Future development could expand the process to show other originating software, however the authors felt that Ansys and Abaqus enveloped most aerospace models seen in industry, sans Nastran.

Introduction and Background

When downstream partners are performing dynamic analyses, it's important to have an accurate representation of all model components. While a FEM of the entire assembly would be ideal, it is typically infeasible due to degrees of freedom (DOF) of the fully integrated model, software incompatibility (e.g., if a payload were modeled in Ansys while the spacecraft is modeled in Nastran), a desire of one partner or the other to not divulge proprietary information, etc. One solution to this scenario is to use a reduced dynamic model, the most common example being the HCB method.

Initially developed by Hurty [1] and expanded on by Craig and Bampton [2], the HCB method is an industry standard component mode synthesis (CMS) method that provides an accurate flexible dynamic representation of a FEM. In addition to potentially significant reduction in degrees of freedom (DOF), an HCB also provides the means of providing a model to downstream partners without divulging design details or proprietary information as it only includes mass, stiffness, and optionally damping and output transformation matrices. Often, launch vehicle providers request HCBs during CLA cycles to plan for upcoming missions and provide spacecraft environments, but HCBs can also be used by spacecrafts to similarly provide launch or test environments to upstream payloads, or as an efficient way of analyzing optical performance, aeroelastic guidance, navigation, and control (GNC), fine robotic arm control, etc.

Historically, HCBs were created using a custom direct matrix abstraction program (DMAP) in Nastran. With the development of the EXTSEOUT case control card, generating an HCB is now a standard output procedure of the normal modes solution. As the industry has expanded, other software packages have become more common place, including Ansys and Abaqus [3, 4]. While Ansys and Abaqus each have CMS capability, including the HCB method, their defaults are their own respective binary file formats. This becomes ineffective when providing the model to downstream partners who may not be performing their work in that software, and so the Nastran OP4 format has become a standard HCB representation format [5]. Neither Ansys or Abaqus support outputting an HCB in the Nastran OP4 format, so this paper details the methodology to produce an HCB in Nastran OP4 format when your FEM is in Abaqus or Ansys.

Matt Griebel · Taylor Wender
Quartus Engineering Incorporated
e-mail: mgriebel@quartus.com; taylor.wender@quartus.com

Kristin B. Zimmerman, PhD (eds.), *Dynamic Substructuring & Transfer Path Analysis, 2026, Vol. 4*, Conference Proceedings of the Society for Experimental Mechanics Series, https://doi.org/10.1007/978-8-743-81443-6_10

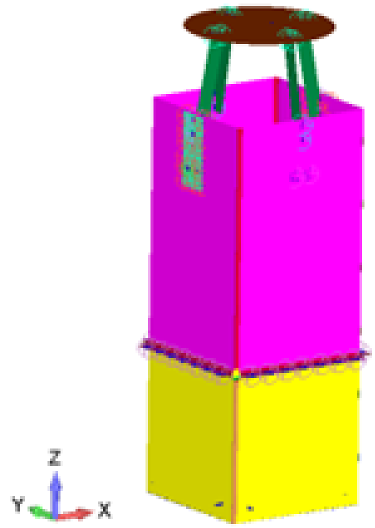

Fig. 1 ShockSat FEM

Methodology

Although this paper is not explicitly concerned with shock analysis or predictions, a simple, publicly available FEM was required. ShockSat (Figure 1) was selected as it has the primary components of most spacecraft, is publicly available, and could easily be converted to both Abaqus and Ansys formats [6]. The conversion from the publicly available Nastran FEM to Abaqus/Ansys FEMs is not shown here, but a comparison of fixed base mode frequencies is shown in Figure 2 to prove the models were accurately converted.

Mode	Fixed Base Mode Frequencies [Hz]		
	Nastran FEM	Ansys FEM	Abaqus FEM
1	42	42	42
2	44	45	45
3	106	109	109
4	112	113	113
5	138	140	139
6	143	144	143
7	157	158	157
8	163	164	164
9	169	170	170
10	170	171	171

Fig. 2 Fixed base mode frequencies for ShockSat FEM in Nastran, Ansys, and Abaqus

To provide an HCB in Nastran OP4 format, standard CMS methods were followed in both Abaqus and Ansys with minimal custom methodology or scripting required (e.g., APDL, DMAP, Python). While converting the FEM itself to Nastran is one solution, converting FEMs is often an arduous and error prone procedure and should be avoided.

In Abaqus, the following steps are taken

1. Perform a two-step substructure generation analysis to produce an Abaqus HCB
   a. Step 1 – Fixed base modal analysis up to the required frequency content of the HCB (*FREQUENCY)
   b. Step 2 – Substructure generation step (*SUBSTRUCTURE GENERATE)
      i. Select the boundary nodes of the previous step as the retained nodal DOF (*RETAINED NODAL DOFS)
      ii. Select all eigenmodes calculated in the previous step (*SELECT EIGENMODES)
      iii. Save the mass and stiffness matrix (*SUBSTRUCTURE MATRIX OUTPUT)
2. Using the generated substructure from the previous analysis, run a matrix generation analysis (*MATRIX GENERATE) and save in the DMIG format using the FORMAT=DMIG parameter on the *MATRIX OUTPUT keyword

In Ansys, a similar procedure is followed

1. Add Substructure Generation to your model tree
   a. Set Analysis Settings options to typical modal analysis settings
   b. Confirm the Matrix Reduction Method is set to Component Mode Synthesis and the Interface Method to Fixed
   c. In the worksheet, define your constraint points
2. Under the Solution in your model tree, add a custom APDL command snippet to read the mass and stiffness matrix from the substructure and export them as DMIGs
   a. It is also prudent to retrieve and write the boundary grids to a CSV file for downstream integration
   b. An example command snippet can be found in the appendix

Now that we have HCBs in DMIG format, conversion to OP4 format is trivial and can be accomplished either using pyYeti or Nastran. Comparison of fixed base modes for all three FEMs and their respective HCBs in OP4 format (analyzed in Nastran) are shown in Figure 3.

Mode	Fixed Base Mode Frequencies [Hz]					
	Nastran		Ansys		Abaqus	
	FEM	HCB	FEM	HCB	FEM	HCB
1	42	42	42	42	42	42
2	44	44	45	45	45	45
3	106	106	109	109	109	109
4	112	112	113	113	113	113
5	138	138	140	140	139	139
6	143	143	144	144	143	143
7	157	157	158	158	157	157
8	163	163	164	164	164	164
9	169	169	170	170	170	170
10	170	170	171	171	171	171

Fig. 3 Fixed base mode frequencies for FEM and HCB models of ShockSat in Nastran, Ansys, and Abaqus. HCB results are analyzed in Nastran using OP4 data converted from the HCB generation steps of the respective software column

Pitfalls, Caveats, and Recommendations

Although the process is relatively straight forward in both Abaqus and Ansys, there are some pitfalls and caveats to be aware of. Standard model checks should be performed on both the FEM and generated HCBs to insure accurate representation. These include free-free modes and grounding, fixed base modes including frequency and modal effective mass comparisons, mass, inertia, and center of gravity comparisons, and potentially frequency response function comparisons.

For Ansys, not all elements and modeling techniques are available for substructure generation, and some that seem to give reasonable results for a normal modes analysis may result in grounding for an HCB created using substructure generation. For most items that are not supported or produce grounding, there is typically a more rudimentary corollary that offers improvement. Typically, things that require Lagrange multipliers (such as joints, contact, etc.) either are not supported or cause grounding and should be replaced with bushings or more accurate interface assumptions for a modal analysis. Additionally, the DMIG exported using the *EXPORT APDL command is not written in symmetric format. Since Nastran expects DMIG in symmetric format, the DMIGs need to be converted before providing them to Nastran. The author's found the easiest way to do this was by reading in the matrices using the publicly available pyYeti Python package and writing the matrices back out as symmetric [7]. This is also a good opportunity to create GRID and SPOINT data since it is not automatically created by Ansys. GRIDs can be read from the file created using the APDL command snippet in the appendix, but SPOINTs will need to be created manually.

For Abaqus, previous versions did not accurately support modal DOF. Newer versions correctly recognize modal DOF and assign them to DOF 0 with an SPOINT during the matrix generation step. Similar grounding issues may be encountered if the model includes Lagrange multiplier modeling assumptions.

For both software, it's important to realize that some small changes may have larger implications that deviate from a true HCB assumption. HCB assumes that all boundary DOF are fixed during model reduction. If any boundary DOF are released, this becomes a "mixed interface" reduction, and if all boundary DOF are released this becomes a "free interface" reduction. The conversion process would ultimately be the same, but the delivered matrices would not be in the expected format and may cause additional issues upon integration. During reduction in Abaqus or Ansys, it is recommended that all interface nodes are fixed, all fixed node DOF are retained, and all calculated modes up to the desired frequency cutoff are retained. Typically, only the nodes that will interface with the downstream model should be retained, but if additional nodes are retained, ensure that sufficient frequency content is captured to accurately represent the model to the desired frequency cutoff. This can usually be verified by monitoring modal effective mass.

If additional nodes are being considered to monitor output at critical locations, an output transformation matrix (OTM) would more appropriate. An OTM transforms results from the boundary and modal DOF to internal results, allowing for efficient calculation of results without expanding the HCB size. The authors did not attempt to create or convert any OTMs and leave that for future development.

Summary and Conclusions

This paper outlines the general process to convert either an Abaqus or Ansys FEM into an HCB in Nastran OP4 format for delivery to downstream partners performing integrated dynamic analysis. With some minor pitfalls and caveats, the process is relatively straight forward and can be accomplished with minimal custom scripting (some minor APDL and Python is required in Ansys). Verification using ShockSat shows that the process produces reliable results and can be followed for more complicated models. It is wise for analysts to perform independent model checks of both the FEM and resulting HCB to ensure that results match expectations and there are no unintended artifacts in the model (e.g., grounding). The authors only considered Abaqus and Ansys as they felt these were the more prominent software used in industry (outside of Nastran), but other software could be included in future development. Similarly, the authors did not consider including any OTMs and leave that for future development.

References

1. W. C. Hurty, "Dynamic analysis of structural systems using component modes," *AIAA Journal 3.4,* pp. 678-685, 1965.
2. R. R. Craig Jr. and M. C. Bampton, "Coupling of substructures for dynamic analyses," *AIAA Journal 6.7,* pp. 1313-1319, 1968.
3. *Abaqus 2024* (2024). Dassault Systemes Simulia Corp.
4. *Ansys 2024 R2* (2024). Ansys, Inc.
5. "Simcenter Nastran DMAP Programmer's Guide," Simcenter Nastran 2406. Siemens Industry Software, Inc. pp. 11-71 – 11-77. 2024.

6. Yunis, S. "ShockSat: A System to Provide Public Data for Advancing Shock Prediction," NESC Academy, https://nescacademy.nasa.gov/workshops/ShockSat/ShockSat1
7. Widrick, T. *pyYeti* (1.4.5), https://pypi.org/project/pyyeti/

Appendix – APDL Command snippet

The following APDL command snippet can be inserted in the solution section of the model tree to output a DMIG and boundary grids from an Ansys generated substructure.

```
FINISH
/filename,dmig_conversion
*DMAT,K80,d,import,sub,file.sub,stiff
*DMAT,M80,d,import,sub,file.sub,mass

*EXPORT,K80,dmig,K80.dmig,,large
*EXPORT,M80,dmig,M80.dmig,,large

FINISH
/CLEAR

/PREP7
ET,50,50
TYPE,50
SE,file

*GET,NUM_NODE,NODE,0,COUNT
*DIM,NODE_DATA,ARRAY,NUM_NODE,7,1
NODE_ID=0
*DO,I,1,NUM_NODE
  NODE_ID=NDNEXT(NODE_ID)
  NODE_DATA(I,1)=NODE_ID
  *GET,NODE_DATA(I,2),NODE,NODE_ID,LOC,X
  *GET,NODE_DATA(I,3),NODE,NODE_ID,LOC,Y
  *GET,NODE_DATA(I,4),NODE,NODE_ID,LOC,Z
  *GET,NODE_DATA(I,5),NODE,NODE_ID,ANG,XY
  *GET,NODE_DATA(I,6),NODE,NODE_ID,ANG,YZ
  *GET,NODE_DATA(I,7),NODE,NODE_ID,ANG,ZX
*ENDDO

*CFOPEN,Nodal_Data,csv
*vwrite,'Node ID','X-Coord','Y-Coord','Z-Coord','XY Angle','YZ Angle','ZX Angle'
%C,%C,%C,%C,%C,%C,%C
*vwrite,node_data(1,1),node_data(1,2),node_data(1,3),node_data(1,4),node_data(1,5),node_data(1,6),node_data(1,7)
%I,%G,%G,%G,%G,%G,%G
*cfclos
```

Appendix – pyYeti Symmetric Matrix Script

The following Python script uses pyYeti to read in an Ansys created DMIG and write out the same DMIG in symmetric format.

```
from pyyeti import nastran
from tkinter import filedialog
```

```
files = filedialog.askopenfilenames(title='Select DMIG Files')
for file in files:
    filename, file_extension = os.path.splitext(file)
    dmig = nastran.rddmig(file,square=True)
    fout = file.replace(file_extension,'_sym'+file_extension)
    nastran.wtdmig(fout,dmig)
```