LASER INTERFEROMETER GRAVITATIONAL WAVE OBSERVATORY

LIGO Laboratory / LIGO Scientific Collaboration

LIGO-T050125-00-D *LIGO* 23 Jul 2005

Transforming Finite Element Eigensolutions to State Space Modal Models

Dennis Coyne

Distribution of this document:
LIGO Scientific Collaboration

This is an internal working note
of the LIGO Project.

California Institute of Technology
LIGO Project – MS 18-34
1200 E. California Blvd.
Pasadena, CA 91125
Phone (626) 395-2129
Fax (626) 304-9834
E-mail: info@ligo.caltech.edu

Massachusetts Institute of Technology
LIGO Project – NW17-161
175 Albany St
Cambridge, MA 02139
Phone (617) 253-4824
Fax (617) 253-7014
E-mail: info@ligo.mit.edu

LIGO Hanford Observatory
P.O. Box 1970
Mail Stop S9-02
Richland WA 99352
Phone 509-372-8106
Fax 509-372-8137

LIGO Livingston Observatory
P.O. Box 940
Livingston, LA 70754
Phone 225-686-3100
Fax 225-686-7189

http://www.ligo.caltech.edu/

1 Introduction

When designing a control system for vibration isolation or pointing control, one generally needs a model of the structural plant. High fidelity models of structural systems can be measured through modal testing or simulated with the finite element technique. Typically the nodal based model that results from an accurate finite element model has too many degrees of freedom to be directly useful in a control system model/synthesis. The order of the finite element model is typically reduced by choosing only the modes of interest and the cardinal points/locations where the response of the system is needed, such as at sensor and actuator locations and the locations of payload elements. A good discussion of the general methods by which a finite element model can be used to create a model appropriate for use in control system development is given in W. Gawronski's text[1].

In this memo, I document the details and Matlab m-files that are used to perform the import of eigensolution results from the I-DEAS finite element code. The method is general and uses industry standard file formats. Extension to other finite element codes in use within LIGO (such as Nastran, Ansys and Algor) should be straightforward, though possibly time consuming if these programs don't generate industry standard file formats.

2 State Space Modal Model

As discussed in W. Gawronski's text, there are several forms of state space modal models. In the following, I used a hybrid modal state space approach in which the system states are modal (displacements and velocities) but the input and outputs are in nodal coordinates. A finite element model of a flexible structure in nodal coordinates can be represented by the following equation:

$$M\ddot{q} + D_0\dot{q} + Kq = B_0u$$

$$y = C_{0q}q + C_{0v}\dot{q}$$

where:

u is the input force vector (length s)

$q = \{x_1, y_1, z_1, Rx_1, Ry_1, Rz_1, x_2, y_2, z_2, Rx_2, Ry_2, Rz_2, ...x_n, y_n, z_n, Rx_n, Ry_n, Rz_n\}$ is the nodal displacement vector for the n nodes of the model (length is n_d = 6 n, the number of degrees of freedom)

$\{x_i, y_i, z_i, Rx_i, Ry_i, Rz_i\}$ are the 6 degrees of freedom for node i

M is the mass matrix (size n_d x n_d)

D_0 is the damping matrix (size n_d x n_d)

K is the stiffness matrix (size n_d x n_d)

B is the input matrix (size n_d x s)

y is the output vector (size r x 1)

[1] W. K. Gawronski, Dynamics and Control of Structures: A Modal Approach, Springer-Verlag, NY, 1998, Chapter 2.

C_{0q} is the output displacement matrix (size r x n_d)

C_{0v} is the output velocity matrix (size r x n_d)

The solution of the eigenproblem,

$$\left(K - \omega^2 M\right)\phi e^{j\omega t} = 0$$

results in n lowest frequency modes, with frequencies $\{\omega_1,\omega_2,\ldots\omega_n\}$, and mode shapes $\{\phi_1,\phi_2,\ldots\phi_n\}$. Defining

$\Omega = diag(\omega_1, \omega_2, ...\omega_n)$, the matrix of natural frequencies

$\Phi = \begin{bmatrix}\phi_1 & \phi_2 & \cdots & \phi_n\end{bmatrix}$, the modal matrix

The modal matrix is used to diagonalize the mass and stiffness matrices, M and K:

$$M_m = \Phi^T M \Phi$$

$$K_m = \Phi^T K \Phi$$

$$Z = \left(1/2\right) M_m^{-1} D_m \Omega^{-1}$$

where the modal damping, D_m, is generally estimated and not transformed from nodal coordinates. Re-casting the governing equation in terms of a state vector,

$$x = \begin{Bmatrix} q \\ \dot{q} \end{Bmatrix}$$

One obtains a state space version of the system:

$$\dot{x} = Ax + Bu$$

$$y = Cx + Du$$

where

$$A = \begin{bmatrix} 0 & I \\ -\Omega^2 & -2Z\Omega \end{bmatrix}$$

$$B = \begin{bmatrix} 0 \\ M_m^{-1}\Phi^T B_0 \end{bmatrix}$$

$$C = \begin{bmatrix} C_{0q}\Phi & C_{0v}\Phi \end{bmatrix}$$

$D = \begin{bmatrix}0\end{bmatrix}$ (here I've chosen to set D to zero, as is typically the case)

Once the eigensolution results (Ω, Φ, M_m) have been read into Matlab, a modal based state space model can be created with the above equations for A, B, C and D.

3 Universal File Formats

The following information is directly from the the University of Cincinnati's Structural Dynamics Research Lab (UC-SDRL), which is leading a consortium of users and vendors to establish and maintain the universal file format for modal analysis data. The first open discussion was held at the 1998 International Modal Analysis Conference (IMAC) in Santa Barbara, CA. More information on the UFF can be found on the UC-SDRL web page.

> Universal File (UF) formats were originally developed by the Structural Dynamics Research Corporation (SDRC) in the late 1960s and early 1970s to facilitate data transfer between computer aided design (CAD) and computer aided test (CAT) in order to facilitate computer aided engineering (CAE). SDRC, as part of EDS, continues to support and utilize the UF formats as part of their CAE software. Currently, MTS, Noise and Vibration Division supports and continues to develop IDEAS`software in the test area that utilizes UF formats.
>
> The formats were originally developed as 80 character (card image), ASCII records that occur in a specific order according to each UF format. As computer files became routinely available, single UF formats were concatentated into computer file structures. Recently, a hybrid UF file structure (UF Dataset 58 Binary) was developed for experimental data that allows data to be stored in a more efficient binary format.
>
> The use of the Universal File Format as a defacto "standard" has been of great value to the experimental dynamics (vibration and acoustic) community, particularly in the area of modal analysis. Both users and vendors have benefited from this defacto standard.

The relevant UFF for transfer function or response spectrum export from I-DEAS for use in developing a Frequency Response Data (FRD) model in Matlab is dataset 58 or dataset 58b.

The relevant UFF for eigensolution export from I-DEAS for use in developing a State Space model in Matlab is dataset 2414. This dataset for "analysis data" does not fall under the purview of the IMAC/UC-SDRL consortium. The format is defined and maintained by SDRC/I-DEAS and can be found in the I-DEAS help CDROM under File Formats Reference Guide. A number of other dataset formats within the Universal file (*.unv) generated by I-DEAS may also be of use in exporting and developing a state space model, such as dataset 151 (header), dataset 164 (units), dataset 2420 (coordinate systems), dataset 2411 (nodes), dataset 2452 (permanent groups) and dataset 609 (eigenvalues and modal mass).

4 Matlab m-files for FEA Results Import

A set of Matlab m-files, readuff.m, are available from the Matlab file exchange[2]. This package currently only reads datasets relevant for modal test results, specifically dataset 151 (header), 15 (coordinate data), 55 (data at nodes), 58 ("measurement" data), 82 (display sequence), 164 (units), and also the hybrid one, 58b ("measurement" data). I have extended this set of m-files to read analysis spectrum data in dataset 58 (ASCI, not the binary version 58b) and eigensolution data (eigenvectors, eigenvalues, modal mass) in dataset 2414. The readuff.m set of m-file functions is not completely universal in that not all options within all datasets are supported. In general an error

[2] Primoz Cermelj, readuff.m, Copyright (c) 2004-2005, beta version 0.9.8b1, Last revision: 06.06.2005

message is intended to indicate where the file read fails, but it is not likely to be robust as yet. This set of m-files is included in the zip-file associated with this document.

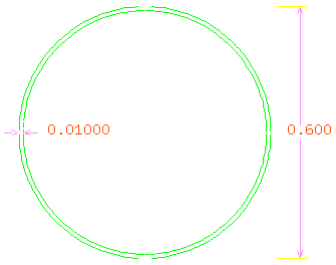
In I-DEAS, in order to generate the response analysis data spectrum (generally a transfer function), one must run a response analysis subsequent to a modal analysis. Then from within the response analysis mode, export the desired response functions in universal file format[3].

An alternative to reading the I-DEAS response spectrum (transfer function) data from UFF dataset 58 is to read the data from a ASCI spreadsheet file format (another I-DEAS output option). This might be similar to the output format of other finite element programs. In fact, an m-file for reading the response analysis data spectrum in this format has been created for use in analysis of the transfer functions for the AL SEI stage 2 with a quadruple pendulum structure attached (as reported in T050014-00).

5 Simple Beam Example

In order to demonstrate the technique and the m-files, a simple cantilevered beam model was exercised as an example. The beam is a zero'th order model of the current (D040519-04) quad suspension structure. The beam is a constant cross-section, thin-walled circular tube with a 2 meter length, fixed at one end and free at the other. The beam material is aluminum and the total mass is 100 Kg. The finite element model is shown in Figure 1 (+X is along the beam axis), the modal frequencies are listed and identified in Table 2 and some of the mode shapes are shown in Figure 2. In Figure 3, the I-DEAS calculated transfer function (tip force to displacement at various positions along the beam) is compared to the I-DEAS imported transfer function and the transfer function calculated from the FRD based on the I-DEAS eigensolution imported into Matlab. Finally, in Figure 4, the transfer function calculated from the state space model is compared to the I-DEAS calculated transfer function. The m-file that created the state space model and the plots, convertFEA2SS.m, is included in Appendix A and in the zip-file associated with this document.

Table 1: Beam Properties




```
No. - name                              : 1 - PIPE 0.6 X 0.010
Element references in active FE model   : 10
Section type                            : Pipe
Dimensions
 Outside diameter          :  0.6
 Wall thickness            :  0.010

Properties
 Area                      :  0.0185354
 Prin. moment of inertia Y :  0.0008067532
 Prin. moment of inertia Z :  0.0008067532
 Shear ratio Y             :  1.883747
 Shear ratio Z             :  1.883747
 Torsional constant        :  0.001613506
 Warping constant          :  0.0
 Warping restraint factor  :  0.0
 Eccentricity Y            :  0.0
 Eccentricity Z            :  0.0
 Plastic modulus Y         :  0.003481334
 Plastic modulus Z         :  0.003481334
 Plastic modulus torsion   :  0.005468479
 Offset rotation angle     :  0.0
 Rt                        :  0.0
 Perimeter                 :  1.884956
 X centroid location       :  0.0
 Y centroid location       :  0.0
 Stress recovery values:
   Pt.    Cy     Cz       Reff
   1     0.3    0.0      0.3
   2     0.0   -0.3      0.3
   3    -0.3    0.0      0.3
   4     0.0    0.3      0.3
```

[3] The option to export the response functions is not available from other analysis modes within I-DEAS, version 9; Response functions are not automatically exported with other finite element data in a "complete" universal file export.

Figure 1: Finite Element Beam Model

using 10 linear, beam elements

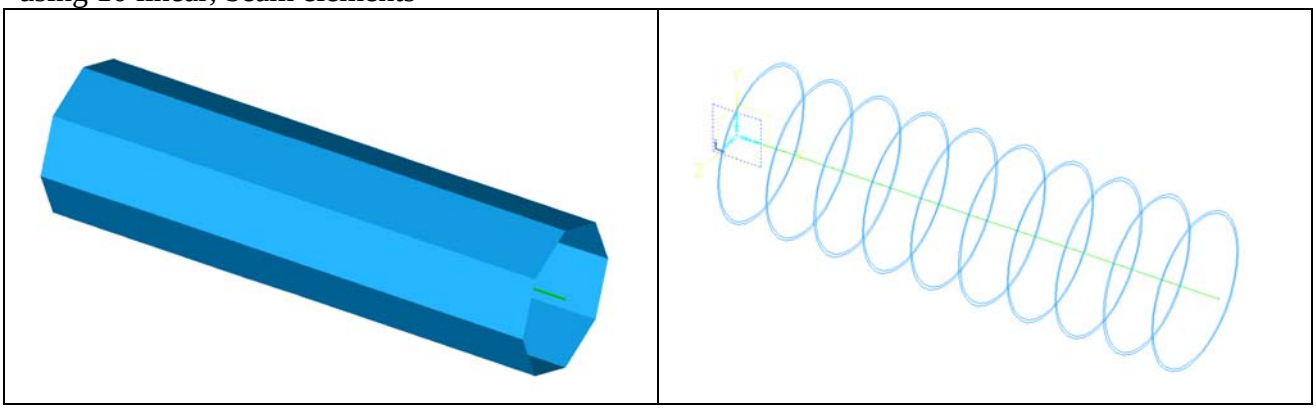
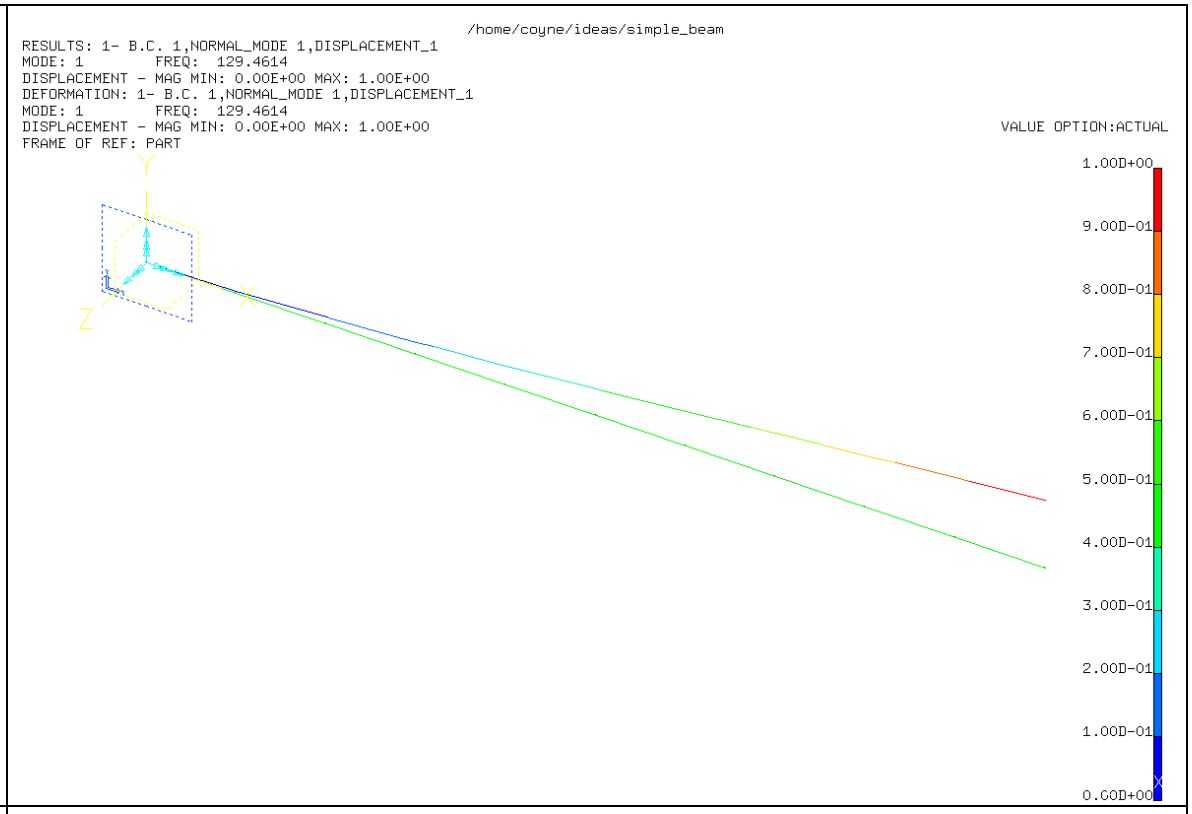
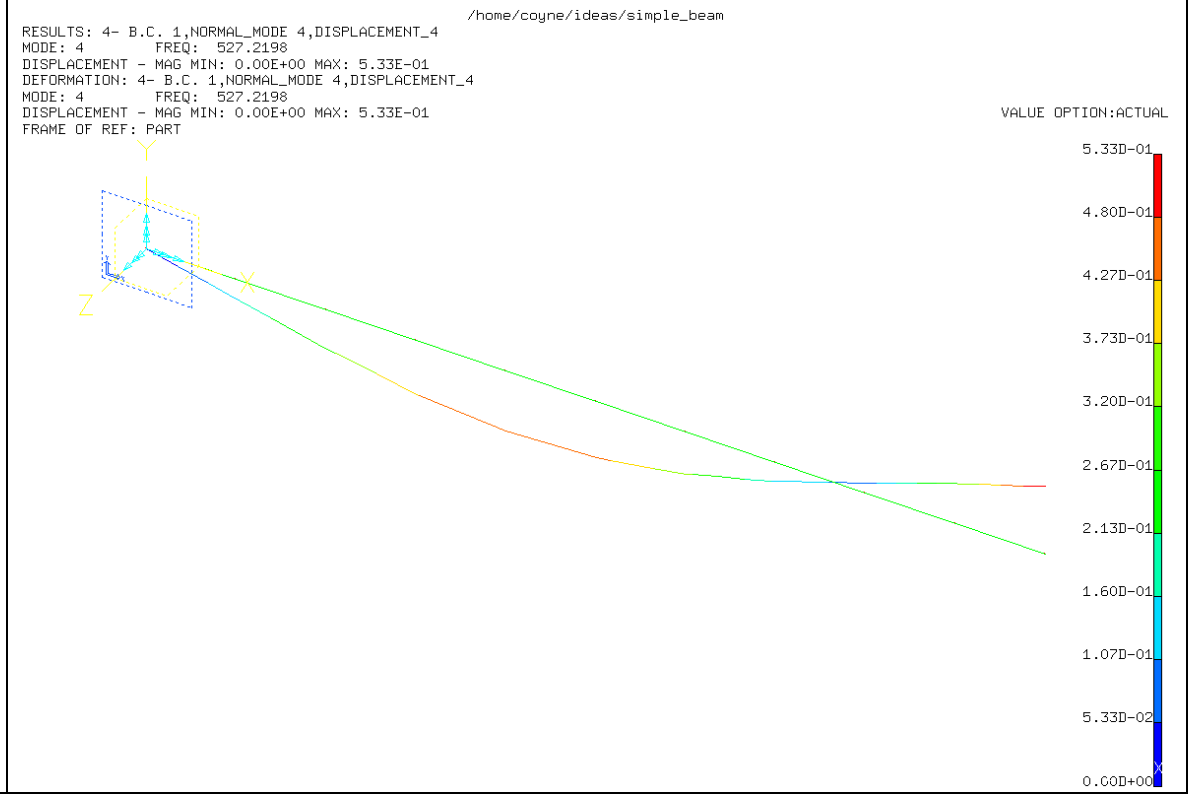


Table 2: Modal Frequencies

	Mode #	Frequency(Hz) Undamped	Modal Prop. Mass	% X-Mass	% Y-Mass	% Z-Mass	--Damping Factors-- % Viscous	% Hysteretic
(1) 1st bending, +Y	*1	129.4614	28.5929	0.00	63.79	0.00	0.00	1.00
(2) 1st bending, +Z	*2	129.4614	28.5929	0.00	0.00	63.79	0.00	1.00
(3) 1st torsional	*3	387.5639	4.33859	0.00	0.00	0.00	0.00	1.00
(4) 2nd bending, +Y	*4	527.2198	12.2405	0.00	21.79	0.00	0.00	1.00
(5) 2nd bending, +Z	*5	527.2198	12.2405	0.00	0.00	21.79	0.00	1.00
(6) 1st axial	*6	632.0976	49.8402	81.06	0.00	0.00	0.00	1.00
(7) 3rd bending, +Y	*7	1127.62	7.43831	0.00	6.05	0.00	0.00	1.00
(8) 3rd bending, +Z	*8	1127.62	7.43831	0.00	0.00	6.05	0.00	1.00
(8) 2nd torsional	*9	1172.27	4.19819	0.00	0.00	0.00	0.00	1.00
(10) 4th bending, +Y	*10	1655.51	3.3206	0.00	1.83	0.00	0.00	1.00

Figure 2: Mode Shapes

1st beam bending mode 129 Hz	
2nd beam bending mode 527 Hz	

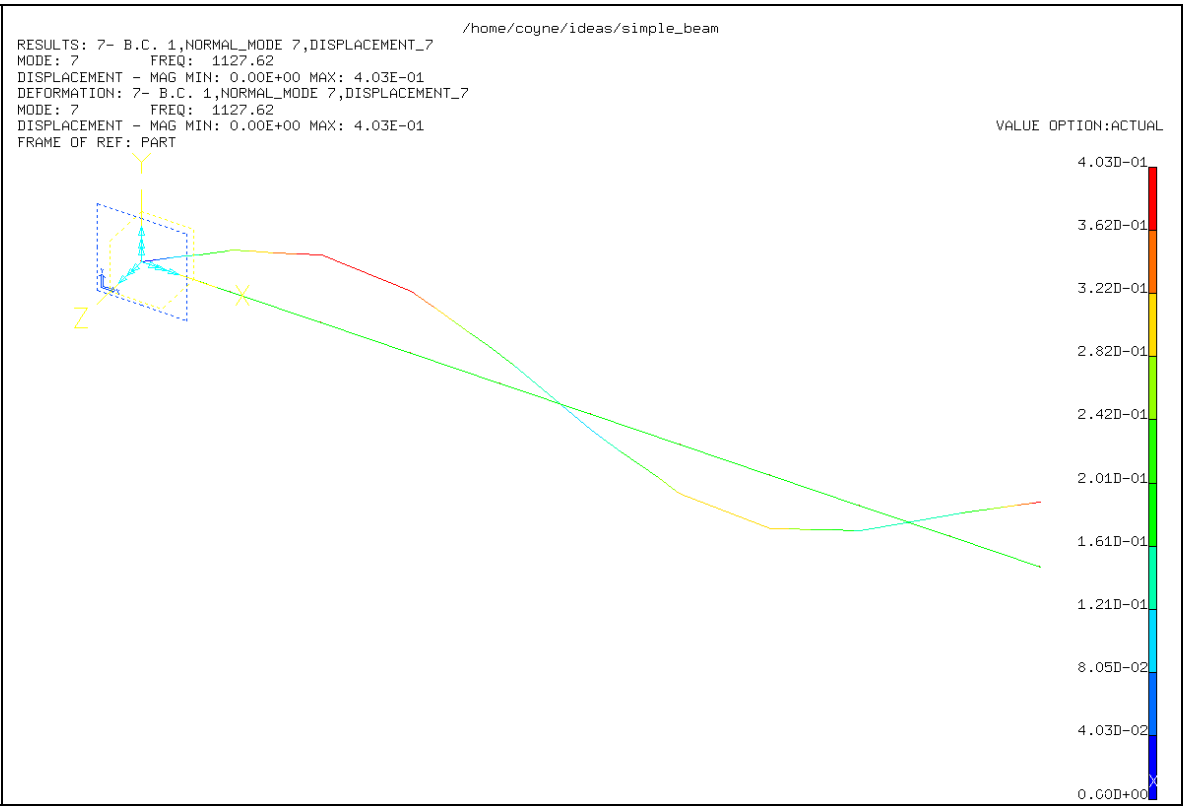
3rd beam bending mode 1127 Hz	

Figure 3: Transfer Functions: Unit Force at Tip to Displacement at Various Positions along the Beam

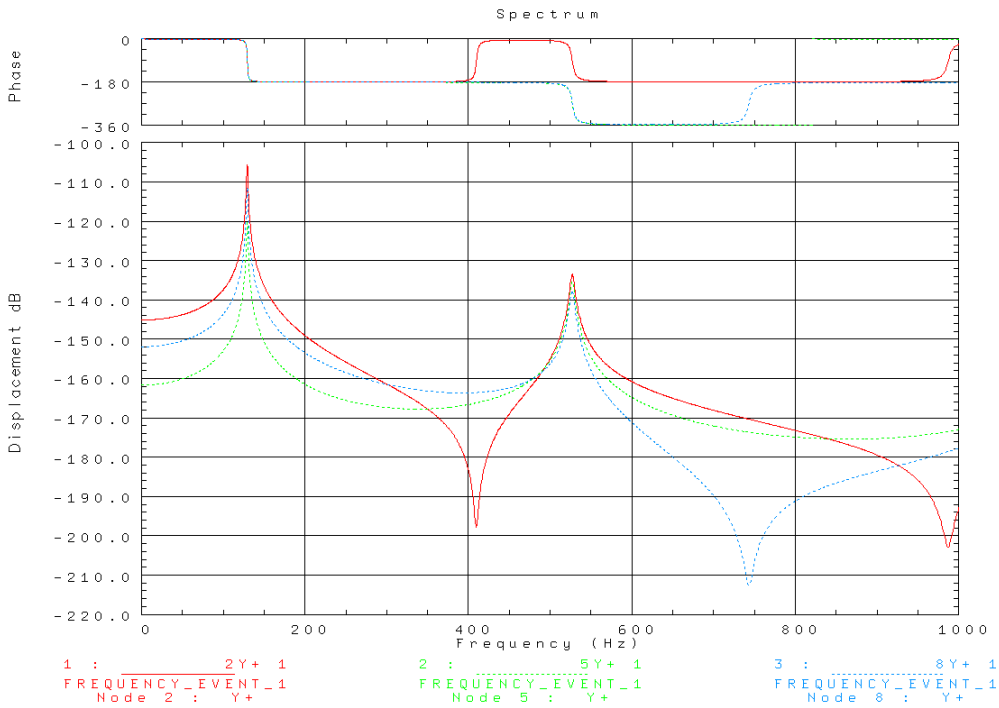
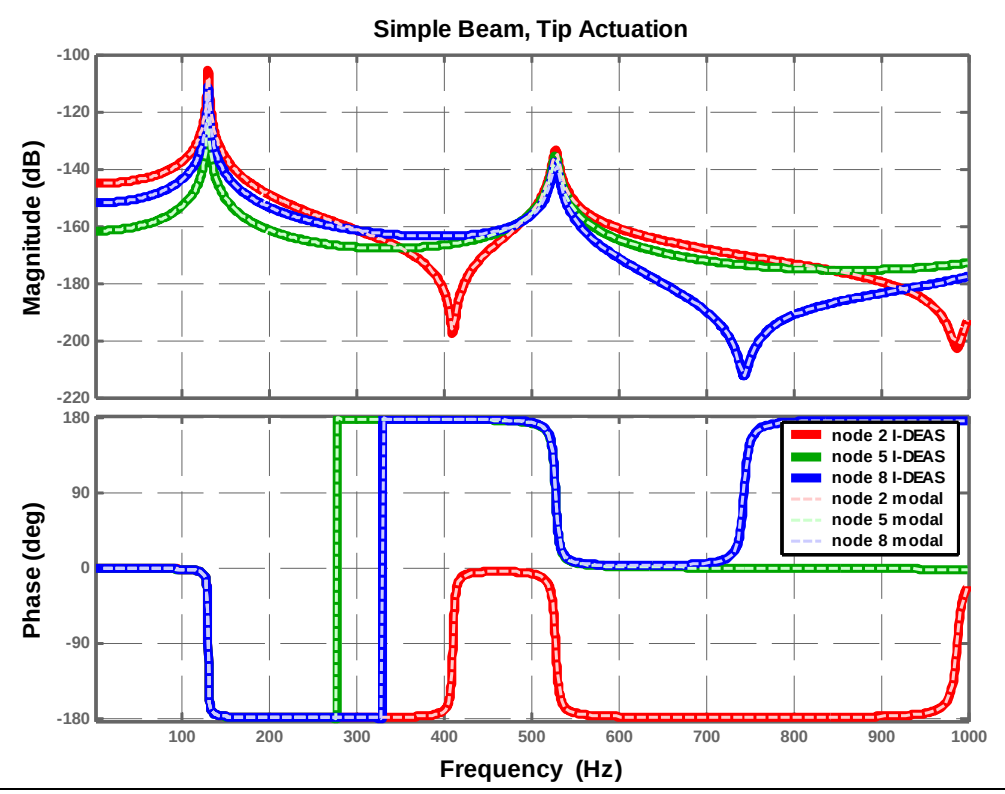
I-DEAS Calculated & Plotted Transfer Function	
Matlab plot of (a) Imported I-DEAS response functions, and (b) transfer function calculated from the eigensolution imported from I-DEAS	

Figure 4: State Space Bode Plot

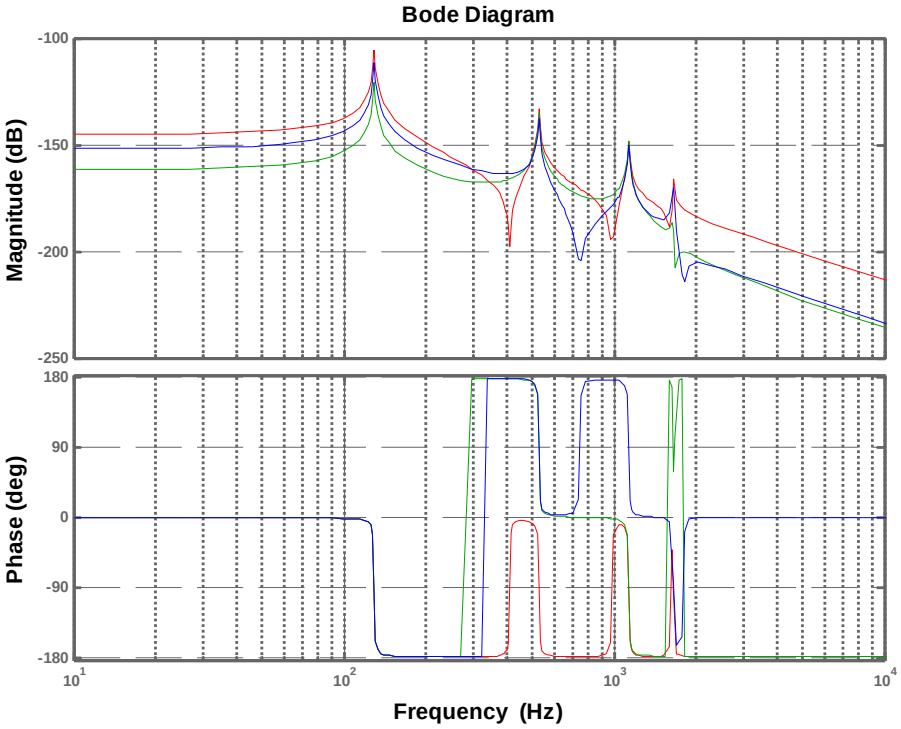
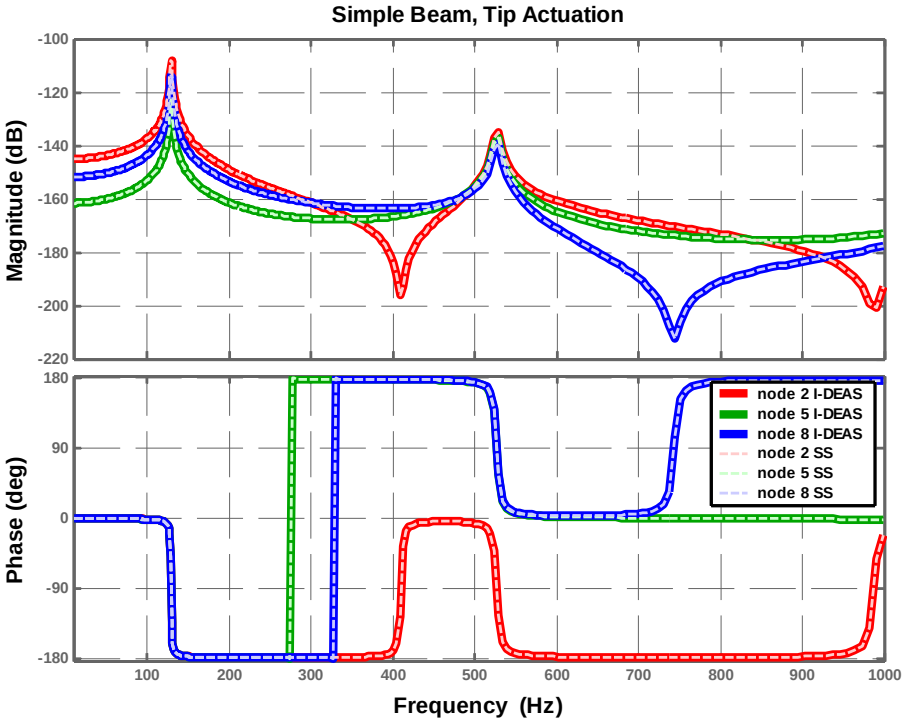


Figure 5: Comparison of Transfer Function calculated from the State Space Model and the I-DEAS Calculated Transfer Function



6 Appendix A: FEA to State Space Matlab m-file

Files included in the zip file associated with this memo.

convertFEA2SS.m	imports modal results from an FEA in dataset2414 format and generates a State Space model (calls uffread.m and import_ideasResponseFunc_dataset58.m to compare results for the "simple beam" example)
import_ideasResponseFunc_dataset58.m	imports response spectrum in dataset58 format (calls uffread.m)
readuff.m	Modified version of Primoz Cermelj's readuff.m, [Copyright (c) 2004-2005, beta version 0.9.8b1, Last revision: 06.06.2005] Includes extension for reading dataset 2414
simple_beam.unv	I-DEAS universal file with FEM and modal results data for the "simple beam" example
tipForce_dataset58.unv	I-DEAS calculated response spectrum for unit force at the beam tip in dataset58 format for the "simple beam" example

A listing of convertFEA2SS.m follows.

```
 1  % convertFEAmodal2SS.m
 2  % Convert Finite Element Analysis (FEA) modal results to State Space Model
 3  % Import Universal File Format (UFF) or UNV file data from I-DEAS
 4  % 2005-07-24, D. Coyne
 5  % Notes:
 6  % 1) Using an extension of the UFF Read m-file, readuff.m
 7  % 2) This version is written for a simple beam example and is not a
 8  % generalization for any finite element model
 9
10  [UffDataSets,Info,errmsg] = readuff('simple_beam2.unv');
11
12  if Info.nErrors ~= 0
13      for ii=1:Info.nErrors
14          disp(Info.errorMsgs{ii});
15      end
16  end
17
18  % echo header information
19  iHeader=find(Info.dsTypes==151);
20  UffDataSets{iHeader}
21
22  % echo units information
23  iUnits=find(Info.dsTypes==164);
24  UffDataSets{iUnits}
25
26  % modal analysis information
27  iModalAnalysis=find(Info.dsTypes==2414);
28  UffDataSets{iModalAnalysis}
29
30  % form the diagonal matrix of natural frequencies, omega
31  nModes = length(iModalAnalysis);
32  freq = zeros(nModes,1);
33  for ii=1:nModes
34      freq(ii) = UffDataSets{iModalAnalysis(ii)}.frequency*2*pi;
35  end
36  omega = diag(freq);
37
38  % check that every mode has the same nodes
39  % SHOULD CHECK THAT THE ORDER OF THE NODAL DATA IS THE SAME AS WELL -- PENDING
40  nodeNumbers = UffDataSets{iModalAnalysis(1)}.nodeNum;
41  nNodes = length(nodeNumbers);
42  for ii=2:nModes
43      if length(UffDataSets{iModalAnalysis(ii)}.nodeNum) ~= nNodes
44          errMessage = ['number of nodes for mode ' num2str(iModalAnalysis(ii).modeNumb↙
    er) ' does not match mode ' num2str(iModalAnalysis(1).modeNumber)];
45          return
46      end
47      if length(setdiff(UffDataSets{iModalAnalysis(ii)}.nodeNum,nodeNumbers)) ~= 0
48          errMessage = ['node numbers for mode ' num2str(iModalAnalysis(ii).modeNumber)↙
```

```
    ' does not match mode ' num2str(iModalAnalysis(1).modeNumber)];
49          return
50      end
51  end
52
53  % form the modal coefficient matrix (matrix of eigenvectors), phi
54  % degree-of-freedom (dof) order is {x1, y1, z1, rx1, ry1, rz1, x2, y2, z2,
55  % rx2, ry2, rz2, x3, ...}
56  % Note: Assuming (hard coded) that ndval = 6 (x, y, z, rx, ry, rz) and not 3 (x, y, z↙
    )
57  nDof = nNodes*6;
58  phi = zeros(nDof,nModes);
59  for ii=1:nModes
60      x = UffDataSets{iModalAnalysis(ii)}.r1;
61      y = UffDataSets{iModalAnalysis(ii)}.r2;
62      z = UffDataSets{iModalAnalysis(ii)}.r3;
63      rx = UffDataSets{iModalAnalysis(ii)}.r4;
64      ry = UffDataSets{iModalAnalysis(ii)}.r5;
65      rz = UffDataSets{iModalAnalysis(ii)}.r6;
66      dofs = [x y z rx ry rz]';
67      dofs = reshape(dofs,prod(size(dofs)),1);
68      phi(:,ii) = dofs;
69  end
70
71  % form the modal mass matrix, Mm
72  modalMasses = zeros(nModes,1);
73  for ii=1:nModes
74      modalMasses(ii) = UffDataSets{iModalAnalysis(ii)}.modalMass;
75  end
76  Mm = diag(modalMasses);
77
78  % form the modal input matrix, Bm
79  % for this example:
80  % u = the force is at the beam tip, node 2, in the y-direction
81  B = zeros(nDof,1);
82  B(1*6+2,1) = 1;
83  Bm = inv(Mm) * transpose(phi) * B;
84
85  % form the modal output matrix, Cm
86  % for this example:
87  % y = y-displacement outputs at nodes 2, 5 and 8
88  % the output velocity matrix, Cv = [0]
89  % the output displacement matrix, Cq = [0,1,0,0,1,0,0,1,0,0,0]
90  nOutputs = 3;
91  Cq = zeros(nOutputs,nDof);
92  Cq(1,1*6+2) = 1;
93  Cq(2,4*6+2) = 1;
94  Cq(3,7*6+2) = 1;
95  Cmq = Cq * phi;
```

```
 96
 97  % modal proportional damping matrix, Z
 98  modalQ = 100;
 99  modalHystereticDampingRatio = 1/(2*modalQ);
100  Z = diag(modalHystereticDampingRatio*ones(nModes,1));
101
102  % Structural Transfer Function Plot
103  % matches the transfer function plot produced by I-DEAS response analysis
104  nFreqs = 500;
105  f = logspace(1,3,nFreqs);
106  w = 2*pi*f;
107  G = zeros(3,nFreqs);
108  % Note: The following structural Transfer Function is eqn. 2.17 from W. Gawronski's
109  % Dynamics and Control of Structures. However, I found it necessary to change sign
110  % on the term (2 i w Z omega) in order to match the direction of phase changes in the
111  % transfer functions from I-DEAS
112  for ii=1:nFreqs
113      G(:,ii) = Cmq * inv(omega^2 - w(ii)^2 * eye(nModes) - 2*sqrt(-1)*w(ii)*Z*omega) * ↙
      Bm;
114  end
115
116  % define a frequency response data model
117  Gsysmodal = frd(G',f,'Units','Hz');
118
119  % Import I-DEAS response data for comparison with the response derived from
120  % the modal imported data
121  [sys] = import_ideasResponseFunc_dataset58('tipForce_dataset58.unv');
122
123  figure;
124  bode(sys(1,1,1),'r',sys(1,1,2),'g',sys(1,1,3),'b',Gsysmodal(1,1,1),'y--',Gsysmodal(1, ↙
     1,2),'w--',Gsysmodal(1,1,3),'m--');
125  sysaxes=findall(gcf,'XScale','log');
126  set(sysaxes,'XScale','linear');
127  title(['Simple Beam, Tip Actuation']);
128  legend('node 2 I-DEAS','node 5 I-DEAS','node 8 I-DEAS','node 2 modal','node 5 modal', ↙
     'node 8 modal');
129  hfig = findall(gcf,'type','line','visible','on');
130  % lets make the linewidth of the I-DEAS calculated transfer functions wider
131  % so that the overlap of the lines is visible and set the color to be
132  % similar but lighter in shade
133  set(hfig,'linewidth',2.0);
134  a = findobj(gcf,'Type','axes');
135  h = findobj(a(2),'Type','line');
136  set(h(4:6),'linewidth',5.0);
137  set(h(1),'Color',[.8 .8 1]);
138  set(h(2),'Color',[.8 1 .8]);
139  set(h(3),'Color',[1 .8 .8]);
140  h = findobj(a(3),'Type','line');
141  set(h(4:6),'linewidth',5.0);
```

```
142  set(h(1),'Color',[.8 .8 1]);
143  set(h(2),'Color',[.8 1 .8]);
144  set(h(3),'Color',[1 .8 .8]);
145
146
147  % Transform into a State Space model
148  % xdot = A x + B u
149  % y = C x + D u
150  A = [zeros(nModes) diag(ones(nModes,1));
151      -omega^2 -2*Z*omega];
152  B = [zeros(nModes,1);
153      Bm];
154  C = [Cmq zeros(3,nModes)];
155  D = zeros(3,1);
156  sysSS = ss(A,B,C,D);
157  size(sysSS)
158  sysSS
159
160  figure;
161  bode(sysSS);
162  bode(sysSS(1,1),'r',sysSS(2,1),'g',sysSS(3,1),'b');
163
164  figure;
165  bode(sys(1,1,1),'r',sys(1,1,2),'g',sys(1,1,3),'b',sysSS(1,1),'y--',sysSS(2,1),'w--',s↙
     ysSS(3,1),'m--',w);
166  sysaxes=findall(gcf,'XScale','log');
167  set(sysaxes,'XScale','linear','XLim',[10 1000]);
168  title(['Simple Beam, Tip Actuation']);
169  legend('node 2 I-DEAS','node 5 I-DEAS','node 8 I-DEAS','node 2 SS','node 5 SS','node ↙
     8 SS');
170  hfig = findall(gcf,'type','line','visible','on');
171  % lets make the linewidth of the I-DEAS calculated transfer functions wider
172  % so that the overlap of the lines is visible and set the color to be
173  % similar but lighter in shade
174  set(hfig,'linewidth',2.0);
175  a = findobj(gcf,'Type','axes');
176  h = findobj(a(2),'Type','line');
177  set(h(4:6),'linewidth',5.0);
178  set(h(1),'Color',[.8 .8 1]);
179  set(h(2),'Color',[.8 1 .8]);
180  set(h(3),'Color',[1 .8 .8]);
181  h = findobj(a(3),'Type','line');
182  set(h(4:6),'linewidth',5.0);
183  set(h(1),'Color',[.8 .8 1]);
184  set(h(2),'Color',[.8 1 .8]);
185  set(h(3),'Color',[1 .8 .8]);
186
187
```