CHAPTER 19

MIMO TWO-STAGE ACTUATOR MODEL

19.1 Introduction

In this chapter we will use an ANSYS model of a two-stage disk drive actuator/suspension system to illustrate the creation of a reduced model for a Multiple Input, Multiple Output (MIMO) system using the balanced reduction method. The results will seem somewhat anticlimactic since the previous chapter covered most aspects of how to use the balanced reduction method. However, understanding the mechanics of setting up a MIMO system should prove useful.

As the track density (tracks per inch, tpi) of disk drives continues to increase, it will be necessary to add a second stage of actuation to the system in order to have the high servo bandwidths required to accurately follow the closely spaced tracks. Many different types of two-stage actuator architectures are being explored. The actuator architecture used for this example is not meant to represent a practical embodiment but will serve to illustrate a two-input, two-output system.

We will begin with descriptions of the actuator system and ANSYS model. Then, ANSYS output, mode shape plots, frequency responses and a partial eigenvector listing will be discussed. The pertinent eigenvector and eigenvalue information will be extracted into a .mat file for input to MATLAB.

The MATLAB code will calculate either dc or peak gains, depending on whether uniform or non-uniform damping is defined. There are four gains to be plotted for this two-input, two-output MIMO system. While dc and peak gains are not required for the "balreal" and "modred" model reduction, they will serve to bridge our understanding from SISO models to MIMO models. We will see the difficulty of choosing which modes to include in a MIMO model using dc or peak gain sorting by discussing the ranking of modes for the four input/output combinations.

In order to perform a balanced reduction, the system is partitioned into rigid body and oscillatory modes, similar to the method used in Chapter 18. The oscillatory modes are balanced and "modred" is used with both the "del" and "mdc" options to reduce the model. Frequency responses for head 0 for both coil and piezo inputs for "del" reduction are shown for various numbers of

reduced modes, from 6 oscillatory states to 20 oscillatory states included. The 20-state case shows both “del” and “mdc” for comparison.

Impulse responses are calculated for oscillatory systems with various numbers of reduced modes retained. The error is plotted as a function of number of modes retained.

19.2 Actuator Description

Figure 19.1 shows top and cross-sectioned side views of the two-stage actuator used for the analysis.

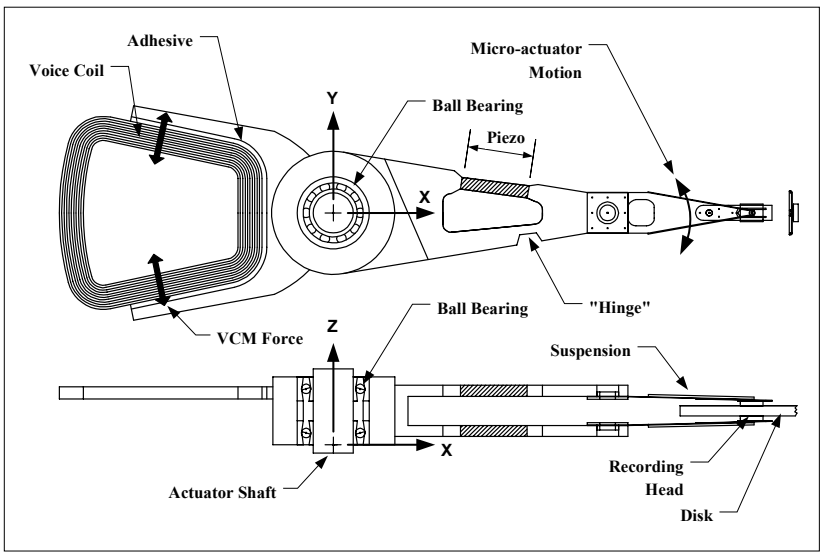

Figure 19.1: Drawing of actuator/suspension system.

The model is similar to the actuator used in Chapters 17 and 18 except that the arms are now the same thickness and are symmetrically located with respect to the pivot bearing z axis centerline. Also, there is now a piezo-actuator bonded into one side of each of the arms. The piezo actuator consists of a ceramic element that changes size when a voltage is applied. In this case, the voltage would be applied to the piezo element so that it changes length, creating a rotation about the “hinge” section in the other side of the arm. This rotation translates the recording head in the circumferential direction. When this “fine positioning” motion is used in conjunction with the VCM’s “coarse positioning” motion, higher servo bandwidths and consequently higher tpi are possible.

The actuator example in the last two chapters had a coil forcing function applied at four nodes in the coil body. Even though there were multiple points at which the force was applied, the fact that the same force was applied to all nodes defined a Single Input system.

Instead of applying voltage as the input into the piezo element, we will assume that we have calculated an equivalent set of forces which can be applied at the ends of the element that will replicate the voltage forcing function. In this model, we will be applying forces to multiple nodes at the ends of both piezo elements. Since the same forces are being applied to both piezo elements, they represent the second input to the now Multi Input system, the first input being the coil force. We will apply equal and opposite forces to the two ends of each piezo actuator, and reverse the signs of the forces applied to the two separate elements. If the same forcing function were applied to both elements, an inertial moment arises which would tend to rotate the entire actuator about the pivot. By using opposite signs for the two arms, this moment is largely eliminated, generating less cross-coupling between the coarse and fine actuator inputs.

In order to make this example a "Multiple Output" system, we will output the displacements of both lower and upper heads, head 0 and head 1.

19.3 ANSYS Model Description

The model description is the same as for the model in Chapter 17. The ANSYS model is shown below, along with a drawing showing the node locations for the coil, piezo elements and heads.

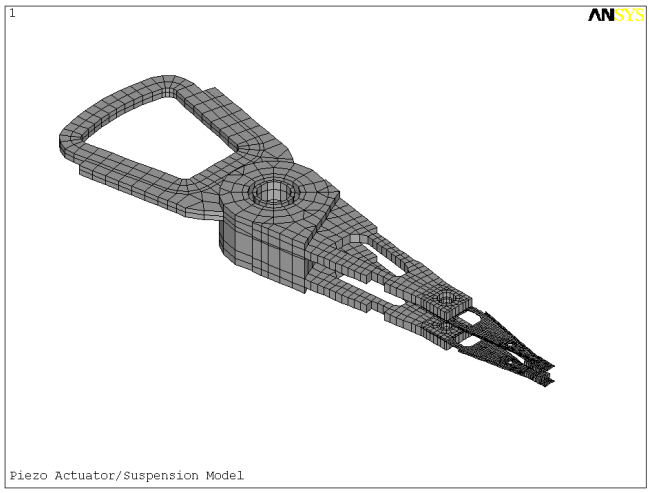

Figure 19.2: Complete piezo actuator/suspension model.

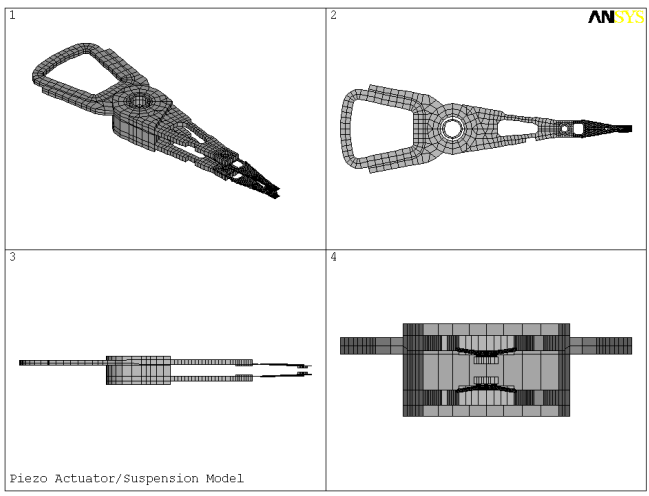

Figure 19.3: Piezo actuator/suspension model, four views.

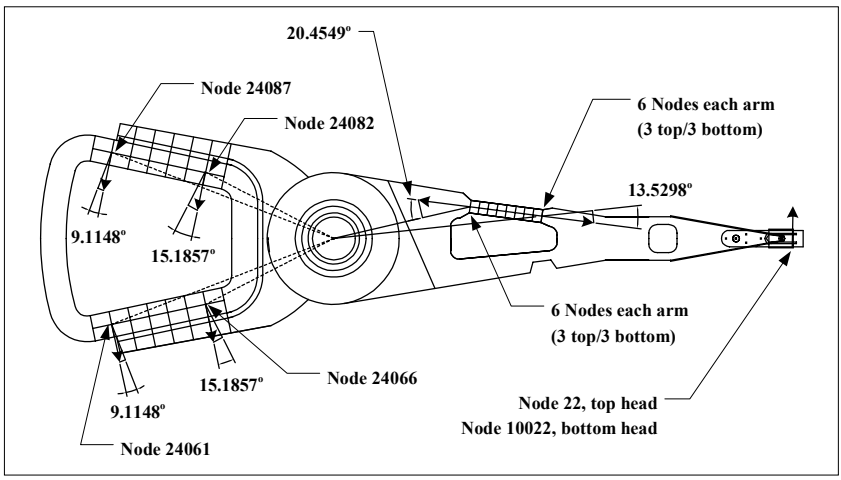

Figure 19.4: Nodes used for reduced MATLAB model, shown with partial mesh at coil and piezo element.

Since the model uses cylindrical coordinates, the coil and piezo forces are at an angle to the radial line joining the pivot bearing centerline to the node location. Both coil and piezo element forces are decomposed into radial and circumferential elements using the angles shown for each in Figure 19.4.

19.4 ANSYS Piezo Actuator/Suspension Model Results

19.4.1 Eigenvalues, Frequency Response

The first 50 modes were extracted using the Block Lanczos method. Frequency versus mode number is plotted in Figure 19.5.

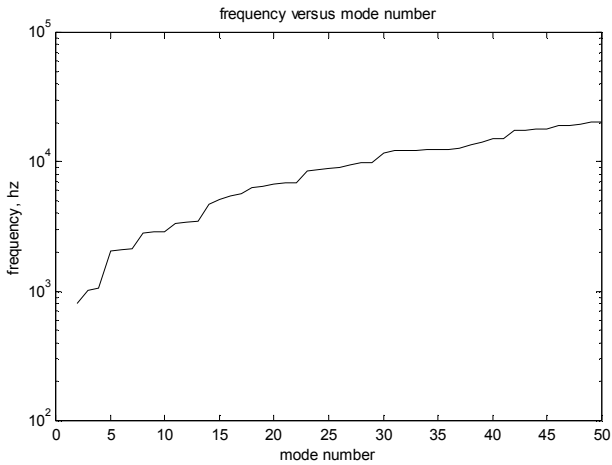

Figure 19.5: Frequencies versus mode number.

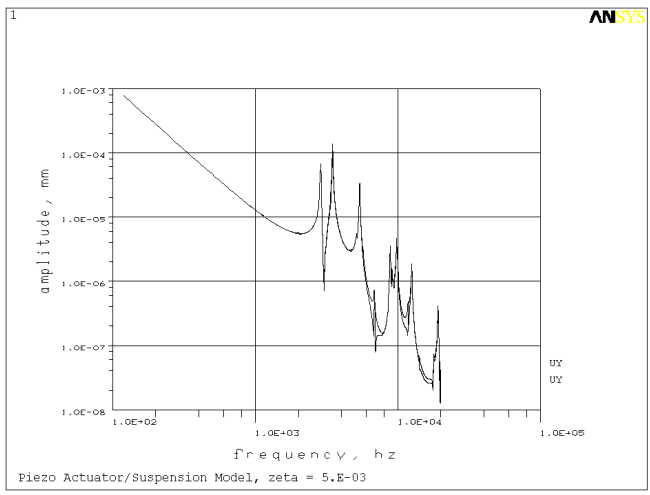

Figure 19.6: Coil input frequency responses for head 0 and head 1 from ANSYS, zeta = 0.005.

Figure 19.6 is the frequency response from ANSYS for coil input for both heads. The same frequency response from the 50-mode MATLAB model is shown in Figure 19.7. Figure 19.8 plots the frequency response for the two piezo inputs.

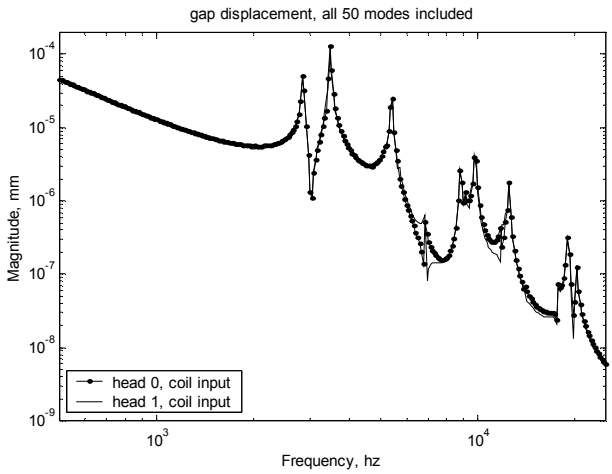

Figure 19.7: Coil input frequency response from MATLAB, zeta = 0.005.

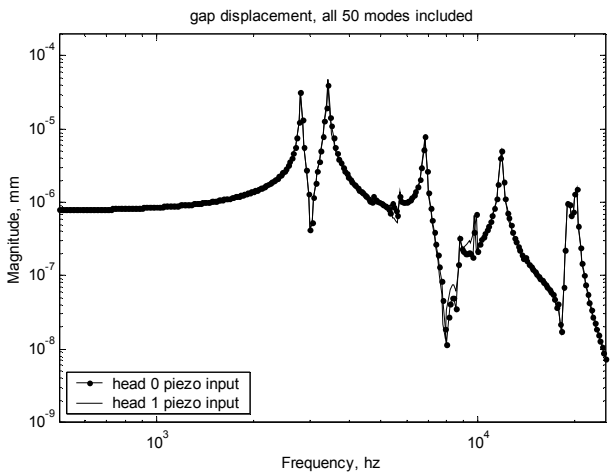

Figure 19.8: Piezo input frequency response from MATLAB, zeta = 0.005.

19.4.2 Mode Shape Plots

Selected mode shape plots are shown below, with a brief discussion of each in the following section.

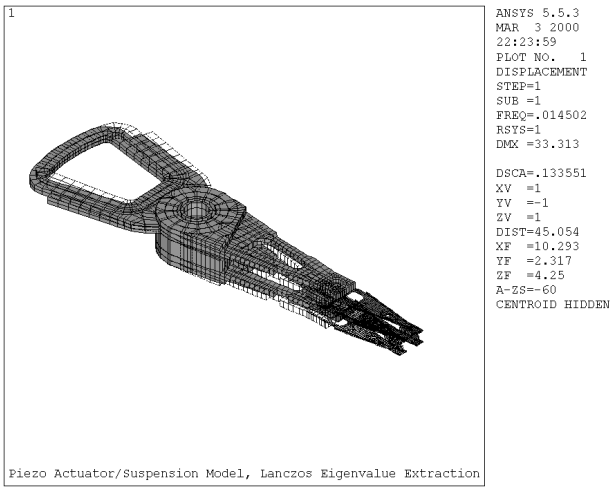

Figure 19.9: Mode 1 undeformed/deformed plot, 0.014 hz, rigid body rotation.

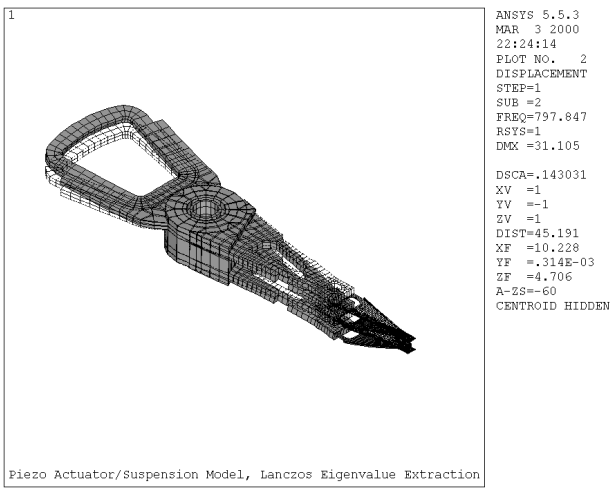

Figure 19.10: Mode 2, 798 hz, actuator pitching mode.

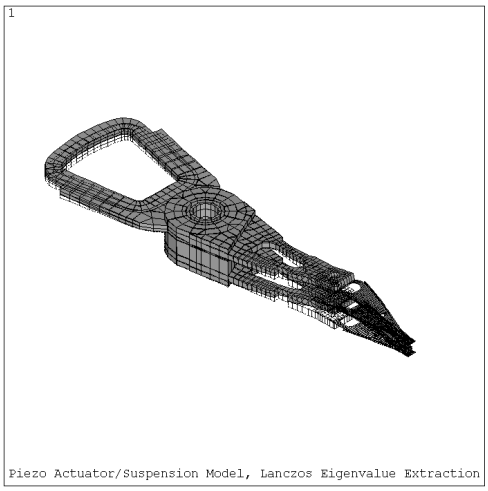

Figure 19.11: Mode 3, 1004 hz, arm/coil bending in phase.

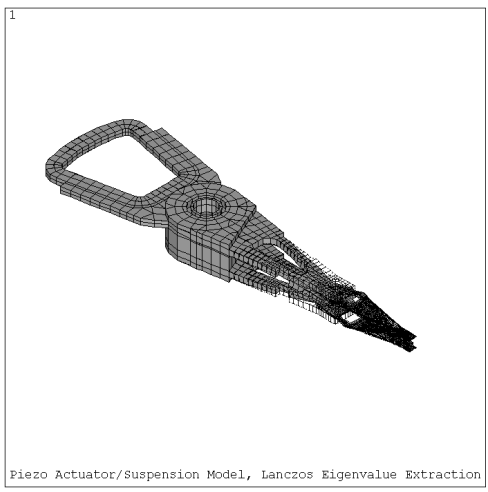

Figure 19.12: Mode 4, 1055 hz, arms bending out of phase.

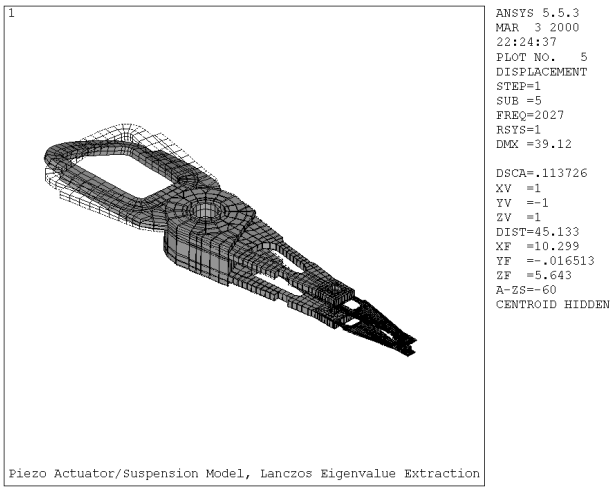

Figure 19.13: Mode 5, 2027 hz, actuator/coil torsion about x axis.

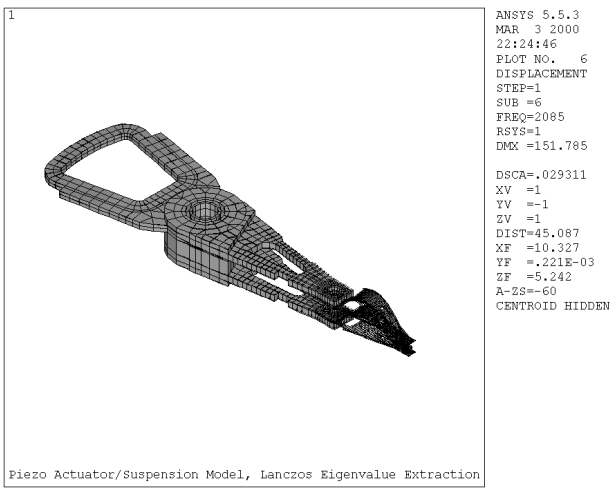

Figure 19.14: Mode 6, 2085 hz, suspension bending mode, some arm interaction.

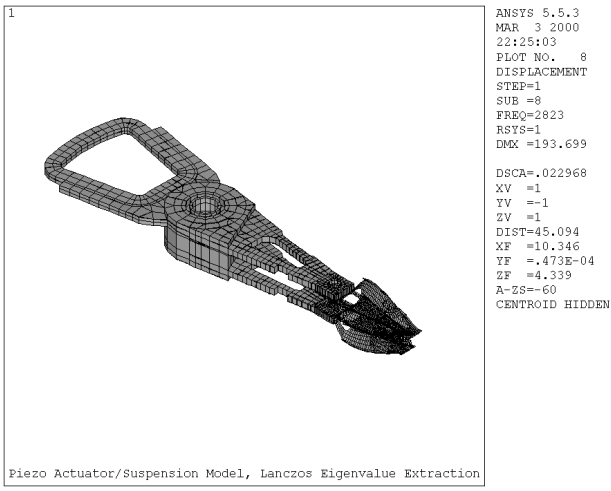

Figure 19.15: Mode 8, 2823 hz, suspension torsion, in phase, arm tip interaction.

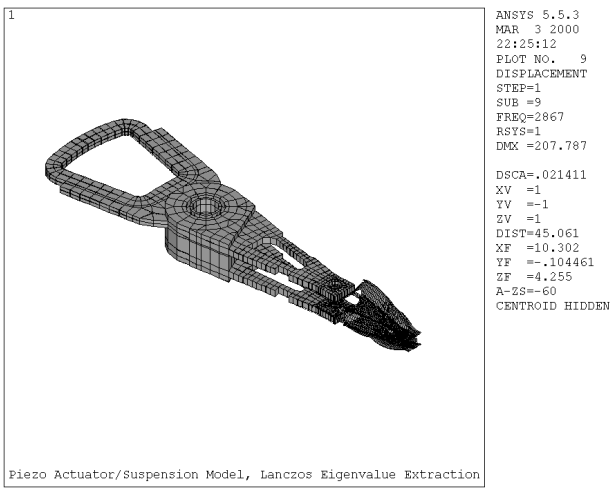

Figure 19.16: Mode 9, 2867 hz, suspension torsion, out of phase.

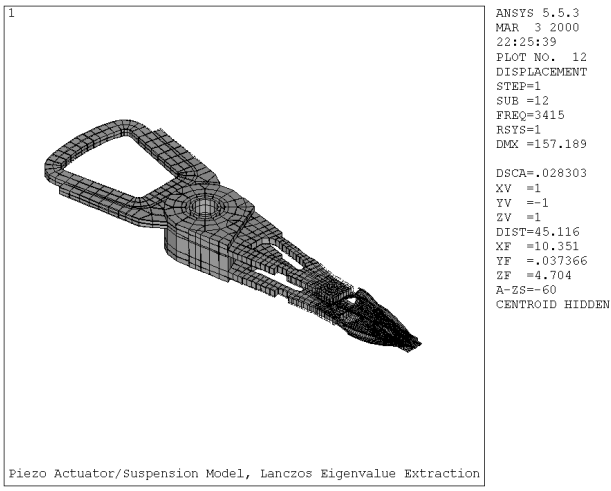

Figure 19.17: Mode 12, 3415 hz, suspension torsion, arm tip lateral.

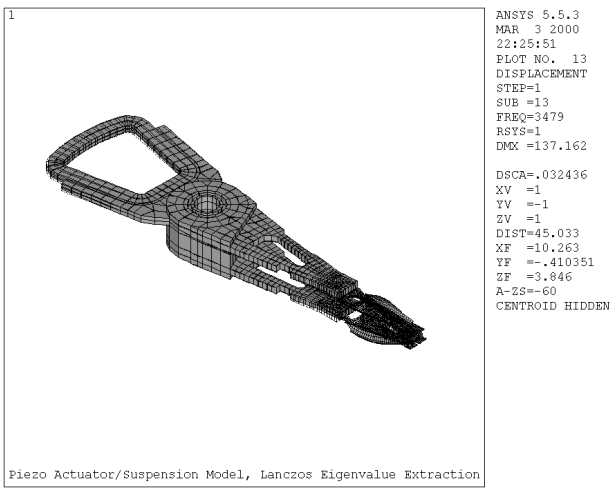

Figure 19.18: Mode 13, 3479 hz, coil/arm/suspension lateral mode.

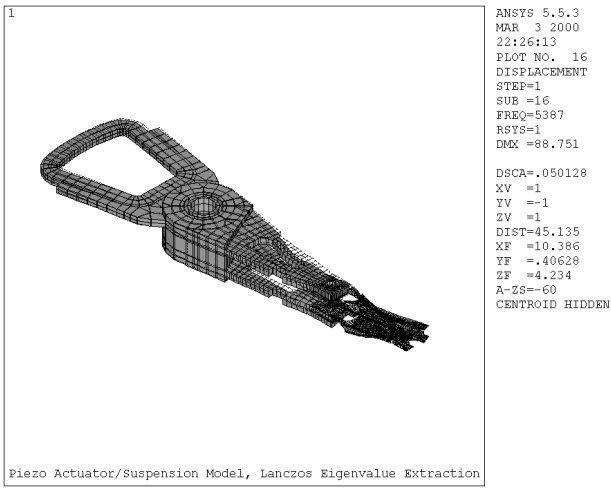

Figure 19.19: Mode 16, 5387 hz, suspension sway, arm tip lateral.

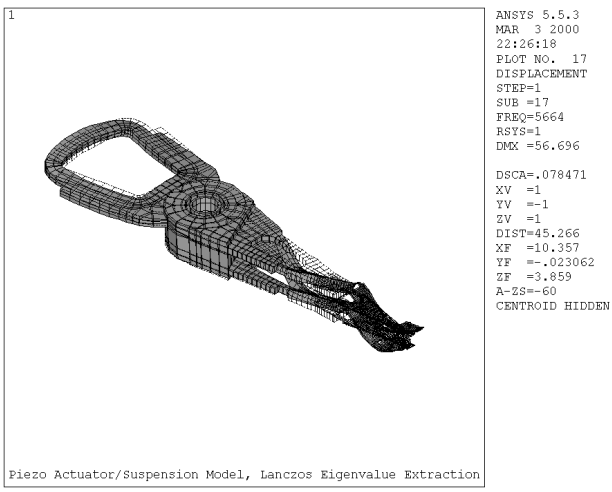

Figure 19.20: Mode 17, 5664 hz, piezo bending, arm tip torsion, coil bending.

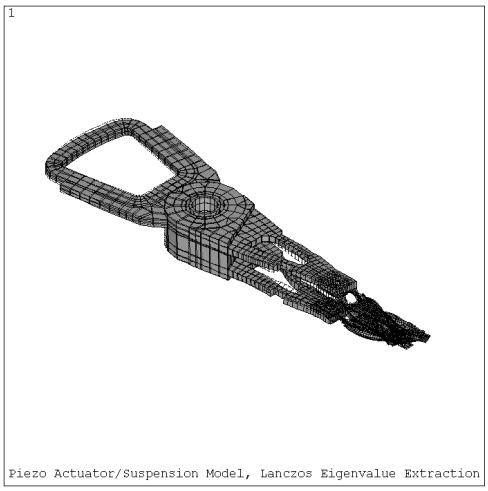

Figure 19.21: Mode 21, 6822 hz, suspension/arm lateral out of phase.

19.4.3 Mode Shape Discussion

As in Chapter 17, we will now describe the major modes which couple into the frequency response as well as several that do not couple, associating them with the frequency responses in Figures 19.7 and 19.8.

Mode 1 is the rigid body rotation mode, which ANSYS again does not calculate at zero hz because of slight geometric and numerical roundoff issues. The frequency for the rigid body mode is set to zero in the MATLAB code.

Modes 2, 3 and 4 are all modes which involve motion only in the x-z plane, bending type motions. Since the motions are perpendicular, or orthogonal, to the direction of input forces and output displacements, they do not couple into any of the frequency responses.

Mode 5 is an actuator/coil torsion mode, rotating about the x axis. A similar mode can be seen on the model in Chapter 17 as a small pole/zero pair on head 1. A torsional mode like this can be excited by: (1) coil forces, since the coil is offset from both the mass center and bearing stiffness center, and (2) inertial forces, because of the asymmetry of the structure about the mass center location in the z direction. Because the arms are more symmetric on this model than the model in Chapter 17, the pole/zero mode does not appear on the frequency response plot of either head. We will see in the dc gain ranking that mode 5 is two orders of magnitude less important than the major

modes of the system for coil input, and is almost three orders of magnitude less important for piezo input.

Mode 6 is a suspension bending mode, once again a bending-only mode with no coupling into the circumferential direction.

Mode 8 is a suspension torsion, arm-tip interaction mode. It is the second most important mode for piezo input, but is unimportant for coil input.

Mode 9 is a suspension torsion mode. It is the second most important mode for coil input, but is unimportant for piezo input. The peak on the two frequency responses, just below 3 khz, is in fact two different frequencies and two different modes for the two different forcing functions. For the coil input the peak is at 2867 hz, mode 9. For piezo input, the peak is at 2823 hz, mode 8.

Modes 12 and 13 are the most important modes for piezo and coil inputs, respectively. Mode 12 involves arm tip lateral motion which the piezo can easily excite. Mode 13 is the "system" lateral mode with all components moving laterally, in phase.

Mode 16, another mode involving the tips of the arms and this time the suspension sway mode, is the third most important mode for coil input.

Mode 17 is the fifth most important piezo excitation mode, involving piezo bending, arm tip torsion and coil bending.

Mode 21 is the third most important mode for piezo excitation, with the suspensions and arms moving laterally, out of phase.

19.4.4 ANSYS Output Listing

The ANSYS output listing for input and output nodes for modes 1, 2 and 13 are listed below. These three modes were selected for discussion in order to highlight different aspects of the eigenvectors. Compared with the ANSYS output listing in Chapter 17, there are significantly more nodes in the output, with the additional nodes representing the six nodes at each end of the bottom and top piezo elements.

The rigid body mode, mode 1, should have only UY displacements (circumferential motion in the cylindrical coordinate system). Mode 2, an actuator pitching mode has its most significant motion in the UZ direction, with some slight coupling into the UX and UY directions. Mode 13 is a highly coupled mode, with significant displacements in all three directions for

some nodes. The UY direction displacements are significant with respect to the UY displacements of mode 2.

***** POST1 NODAL DEGREE OF FREEDOM LISTING *****

LOAD STEP= 1 **SUBSTEP= 1**
FREQ= 0.14502E-01 LOAD CASE= 0

THE FOLLOWING DEGREE OF FREEDOM RESULTS ARE IN COORDINATE SYSTEM 1

NODE	**UX**	**UY**	UZ	ROTX	ROTY	ROTZ
22	0.30584E-06	32.618	0.11285E-11	0.0000	0.0000	0.0000
10022	0.30627E-06	32.618	-0.46777E-10	0.0000	0.0000	0.0000
21538	0.85322E-07	9.7742	0.21745E-08	0.0000	0.0000	0.0000
21546	0.82634E-07	14.735	0.36557E-08	0.0000	0.0000	0.0000
21576	0.10309E-06	9.9634	0.21924E-08	0.0000	0.0000	0.0000
21584	0.16887E-06	14.883	0.37407E-08	0.0000	0.0000	0.0000
21617	0.10951E-06	10.147	0.22079E-08	0.0000	0.0000	0.0000
21625	0.11092E-06	14.978	0.37980E-08	0.0000	0.0000	0.0000
22538	0.85184E-07	9.7742	0.21706E-08	0.0000	0.0000	0.0000
22546	0.82327E-07	14.735	0.36546E-08	0.0000	0.0000	0.0000
22576	0.10295E-06	9.9634	0.21900E-08	0.0000	0.0000	0.0000
22584	0.16856E-06	14.883	0.37381E-08	0.0000	0.0000	0.0000
22617	0.10937E-06	10.147	0.22067E-08	0.0000	0.0000	0.0000
22625	0.11061E-06	14.978	0.37940E-08	0.0000	0.0000	0.0000
24061	0.11911E-06	16.888	-0.95894E-09	0.0000	0.0000	0.0000
24066	0.77030E-07	10.226	-0.53758E-09	0.0000	0.0000	0.0000
24082	0.68150E-07	10.226	-0.48785E-09	0.0000	0.0000	0.0000
24087	0.10037E-06	16.888	-0.86954E-09	0.0000	0.0000	0.0000
24538	0.84850E-07	9.7742	0.20872E-08	0.0000	0.0000	0.0000
24546	0.81937E-07	14.735	0.18321E-08	0.0000	0.0000	0.0000
24576	0.10262E-06	9.9634	0.20998E-08	0.0000	0.0000	0.0000
24584	0.16817E-06	14.883	0.17648E-08	0.0000	0.0000	0.0000
24617	0.10904E-06	10.147	0.21122E-08	0.0000	0.0000	0.0000
24625	0.11021E-06	14.978	0.17139E-08	0.0000	0.0000	0.0000
25538	0.84745E-07	9.7742	0.20835E-08	0.0000	0.0000	0.0000
25546	0.82082E-07	14.735	0.18310E-08	0.0000	0.0000	0.0000
25576	0.10251E-06	9.9634	0.20975E-08	0.0000	0.0000	0.0000
25584	0.16832E-06	14.883	0.17623E-08	0.0000	0.0000	0.0000
25617	0.10894E-06	10.147	0.21110E-08	0.0000	0.0000	0.0000
25625	0.11036E-06	14.978	0.17100E-08	0.0000	0.0000	0.0000

MAXIMUM ABSOLUTE VALUES

NODE	10022	22	21625	0	0	0
VALUE	0.30627E-06	32.618	0.37980E-08	0.0000	0.0000	0.0000

*ENDDO INDEX= I

***** POST1 NODAL DEGREE OF FREEDOM LISTING *****

LOAD STEP= 1 **SUBSTEP= 2**
FREQ= 797.85 LOAD CASE= 0

THE FOLLOWING DEGREE OF FREEDOM RESULTS ARE IN COORDINATE SYSTEM 1

NODE	**UX**	**UY**	UZ	ROTX	ROTY	ROTZ
22	0.49229	-0.14022	-0.10321E-03	0.0000	0.0000	0.0000
10022	-0.89140	0.14245	-0.83465E-03	0.0000	0.0000	0.0000
21538	-1.0283	0.18631	-4.0091	0.0000	0.0000	0.0000
21546	-1.5471	0.23464E-01	-10.200	0.0000	0.0000	0.0000
21576	-1.0204	0.23663	-4.0561	0.0000	0.0000	0.0000
21584	-1.5459	0.72962E-01	-10.473	0.0000	0.0000	0.0000
21617	-1.0084	0.27685	-4.0950	0.0000	0.0000	0.0000
21625	-1.5436	0.11594	-10.631	0.0000	0.0000	0.0000
22538	-0.61275	0.10972	-4.0090	0.0000	0.0000	0.0000
22546	-0.12481	0.83127E-01	-10.200	0.0000	0.0000	0.0000
22576	-0.60478	0.13415	-4.0560	0.0000	0.0000	0.0000
22584	-0.12184	0.86554E-01	-10.473	0.0000	0.0000	0.0000
22617	-0.60100	0.15502	-4.0950	0.0000	0.0000	0.0000
22625	-0.11925	0.89513E-01	-10.631	0.0000	0.0000	0.0000
24061	-0.35220	0.13939	19.652	0.0000	0.0000	0.0000
24066	-0.33572	0.17431	7.3143	0.0000	0.0000	0.0000
24082	-0.33512	-0.17241	7.3089	0.0000	0.0000	0.0000
24087	-0.35171	-0.13563	19.644	0.0000	0.0000	0.0000
24538	0.22023	-0.36868E-01	-4.0205	0.0000	0.0000	0.0000
24546	-0.27795	-0.52244E-01	-10.250	0.0000	0.0000	0.0000
24576	0.21597	-0.43317E-01	-4.0680	0.0000	0.0000	0.0000
24584	-0.27997	-0.42854E-01	-10.524	0.0000	0.0000	0.0000
24617	0.21591	-0.49478E-01	-4.1074	0.0000	0.0000	0.0000
24625	-0.28139	-0.34705E-01	-10.683	0.0000	0.0000	0.0000
25538	0.63806	-0.11349	-4.0206	0.0000	0.0000	0.0000
25546	1.1532	0.79337E-02	-10.250	0.0000	0.0000	0.0000
25576	0.63387	-0.14598	-4.0680	0.0000	0.0000	0.0000
25584	1.1531	-0.29036E-01	-10.524	0.0000	0.0000	0.0000
25617	0.62557	-0.17161	-4.1074	0.0000	0.0000	0.0000
25625	1.1519	-0.61159E-01	-10.683	0.0000	0.0000	0.0000

MAXIMUM ABSOLUTE VALUES

NODE	21546	21617	24061	0	0	0
VALUE	-1.5471	0.27685	19.652	0.0000	0.0000	0.0000

***** POST1 NODAL DEGREE OF FREEDOM LISTING *****

LOAD STEP= 1 **SUBSTEP= 13**
FREQ= 3479.3 LOAD CASE= 0

THE FOLLOWING DEGREE OF FREEDOM RESULTS ARE IN COORDINATE SYSTEM 1

NODE	**UX**	**UY**	UZ	ROTX	ROTY	ROTZ
22	-2.1984	60.376	-0.14239E-02	0.0000	0.0000	0.0000
10022	-1.9960	77.045	0.31840E-01	0.0000	0.0000	0.0000
21538	0.80764E-01	0.40397E-01	0.49848	0.0000	0.0000	0.0000
21546	-6.4836	3.9912	-1.2673	0.0000	0.0000	0.0000
21576	0.72358E-01	0.63009E-01	0.42663	0.0000	0.0000	0.0000
21584	-7.6689	4.6553	-1.8884	0.0000	0.0000	0.0000
21617	0.12273	0.57379E-01	0.37047	0.0000	0.0000	0.0000

```
 21625    -8.7016          5.1772        -2.4325      0.0000      0.0000      0.0000
 22538     0.87706E-01     0.17543        0.56748     0.0000      0.0000      0.0000
 22546    -6.2831          5.0182        -1.2225      0.0000      0.0000      0.0000
 22576     0.92974E-01     0.18824        0.48659     0.0000      0.0000      0.0000
 22584    -7.4322          5.6835        -1.8299      0.0000      0.0000      0.0000
 22617     0.14368         0.17617        0.42076     0.0000      0.0000      0.0000
 22625    -8.4357          6.2048        -2.3541      0.0000      0.0000      0.0000
 24061    -1.9369        -12.670         -0.95604     0.0000      0.0000      0.0000
 24066    -1.0801         -4.7937        -1.0649      0.0000      0.0000      0.0000
 24082     1.5007         -4.5559        -1.4595      0.0000      0.0000      0.0000
 24087     2.3829        -12.467          0.10330     0.0000      0.0000      0.0000
 24538    -0.93404E-01     0.37757        1.0909      0.0000      0.0000      0.0000
 24546    -5.5118          4.1576        2.6594       0.0000      0.0000      0.0000
 24576    -0.66009E-01     0.38853        1.0874      0.0000      0.0000      0.0000
 24584    -6.3981          4.6967        3.0133       0.0000      0.0000      0.0000
 24617    -0.78948E-02     0.37908        1.0812      0.0000      0.0000      0.0000
 24625    -7.1715          5.1206        3.3430       0.0000      0.0000      0.0000
 25538    -0.30931         0.47682        1.1451      0.0000      0.0000      0.0000
 25546    -5.2283          3.4392        2.6949       0.0000      0.0000      0.0000
 25576    -0.28463         0.50756        1.1349      0.0000      0.0000      0.0000
 25584    -6.1405          3.9607        3.0595       0.0000      0.0000      0.0000
 25617    -0.21671         0.51131        1.1213      0.0000      0.0000      0.0000
 25625    -6.9354          4.3710         3.4049      0.0000      0.0000      0.0000

MAXIMUM ABSOLUTE VALUES
NODE    21625     10022     25625      0          0          0
VALUE   -8.7016   77.045    3.4049     0.0000     0.0000     0.0000
```

The eigenvalues and eigenvectors are stripped out of the ANSYS actrlpz.eig file and are stored in the MATLAB .mat file actrlpz_eig.mat.

19.5 MATLAB Model, MATLAB Code act8pz.m Listing and Results

19.5.1 Input, dof Definition

The **act8pz.m** MATLAB code starts by defining the degrees of freedom, nodes, directions and locations for the problem for reference in building the model. The degrees of freedom are extracted from the ANSYS eigenvalue/eigenvector listing and are ordered by node number, first the UX direction and then the UY direction. Once again, the UX direction information is required to transform the coil and piezo forces into cylindrical coordinates. The eigenvalue/eigenvector information is then loaded by reading the .mat file **actrlpz_eig.mat** and the rigid body mode is set to zero frequency.

```
%       act8pz.m

        clear all;
```

```
            hold off;

            clf;

%           load the Block Lanczos .mat file actrl_eig.mat, containing evr – the
%           modal matrix, freqvec -the frequency vector and node_numbers - the
%           vector of node numbers for the modal matrix

%           the output for the ANSYS run is the following dof's

%  dof      node     dir          where
%
%  1          22     ux - radial, top head gap
%  2       10022     ux - radial, bottom head gap
%  3       21538     ux - radial, bottom arm piezo, hub end
%  4       21546     ux - radial, bottom arm piezo, head end
%  5       21576     ux - radial, bottom arm piezo, hub end
%  6       21584     ux - radial, bottom arm piezo, head end
%  7       21617     ux - radial, bottom arm piezo, hub end
%  8       21625     ux - radial, bottom arm piezo, head end
%  9       22538     ux - radial, bottom arm piezo, hub end
% 10       22546     ux - radial, bottom arm piezo, head end
% 11       22576     ux - radial, bottom arm piezo, hub end
% 12       22584     ux - radial, bottom arm piezo, head end
% 13       22617     ux - radial, bottom arm piezo, hub end
% 14       22625     ux - radial, bottom arm piezo, head end
% 15       24061     ux - radial, bottom arm piezo, coil
% 16       24066     ux - radial, bottom arm piezo, coil
% 17       24082     ux - radial, bottom arm piezo, coil
% 18       24087     ux - radial, bottom arm piezo, coil
% 19       24538     ux - radial, top arm piezo, hub end
% 20       24546     ux - radial, top arm piezo, head end
% 21       24576     ux - radial, top arm piezo, hub end
% 22       24584     ux - radial, top arm piezo, head end
% 23       24617     ux - radial, top arm piezo, hub end
% 24       24625     ux - radial, top arm piezo, head end
% 25       25538     ux - radial, top arm piezo, hub end
% 26       25546     ux - radial, top arm piezo, head end
% 27       25576     ux - radial, top arm piezo, hub end
% 28       25584     ux - radial, top arm piezo, head end
% 29       25617     ux - radial, top arm piezo, hub end
% 30       25625     ux - radial, top arm piezo, head end
% 31          22     uy - circumferential, top head gap
% 32       10022     uy - circumferential, bottom head gap
% 33       21538     uy - circumferential, bottom arm piezo, hub end
% 34       21546     uy - circumferential, bottom arm piezo, head end
% 35       21576     uy - circumferential, bottom arm piezo, hub end
% 36       21584     uy - circumferential, bottom arm piezo, head end
% 37       21617     uy - circumferential, bottom arm piezo, hub end
% 38       21625     uy - circumferential, bottom arm piezo, head end
% 39       22538     uy - circumferential, bottom arm piezo, hub end
% 40       22546     uy - circumferential, bottom arm piezo, head end
% 41       22576     uy - circumferential, bottom arm piezo, hub end
% 42       22584     uy - circumferential, bottom arm piezo, head end
% 43       22617     uy - circumferential, bottom arm piezo, hub end
```

```
%  44      22625    uy - circumferential, bottom arm piezo, head end
%  45      24061    uy - circumferential, bottom arm piezo, coil
%  46      24066    uy - circumferential, bottom arm piezo, coil
%  47      24082    uy - circumferential, bottom arm piezo, coil
%  48      24087    uy - circumferential, bottom arm piezo, coil
%  49      24538    uy - circumferential, top arm piezo, hub end
%  50      24546    uy - circumferential, top arm piezo, head end
%  51      24576    uy - circumferential, top arm piezo, hub end
%  52      24584    uy - circumferential, top arm piezo, head end
%  53      24617    uy - circumferential, top arm piezo, hub end
%  54      24625    uy - circumferential, top arm piezo, head end
%  55      25538    uy - circumferential, top arm piezo, hub end
%  56      25546    uy - circumferential, top arm piezo, head end
%  57      25576    uy - circumferential, top arm piezo, hub end
%  58      25584    uy - circumferential, top arm piezo, head end
%  59      25617    uy - circumferential, top arm piezo, hub end
%  60      25625    uy - circumferential, top arm piezo, head end

          load actrlpz_eig;

          [numdof,num_modes_total] = size(evr);

          freqvec(1) = 0;          % set rigid body mode to zero frequency

          xn = evr;
```

19.5.2 Forcing Function Definition, dc Gain Calculations

The unity coil force is equally divided between the four coil nodes. For this model, the piezo force, "fpz," is arbitrarily set at 0.2, to be applied with equal magnitudes and with opposite signs to the two ends of each piezo element. For an actual system, the piezo force would be related to the coil force by the appropriate force constants for the VCM and the appropriate voltage/force relationships for the piezo, and would not be arbitrarily chosen.

Given the directions of the coil and piezo forces in Figure 19.4, the forces are transformed to cylindrical coordinates and two forcing function vectors are formed, one for the coil and one for the piezo.

The user is prompted for whether uniform or non-uniform damping is to be used and then dc or peak gains are calculated, respectively.

For a SISO system, we can rank the relative importance of modes using two methods, by using dc or peak gains and by using balancing. For a MIMO system, balancing is the only practical option. However, we will still calculate the dc gains for this MIMO system to get a feel for the relative importance of

each of the modes for both forcing functions. This will require calculating dc gains for the four combinations possible for the two-input, two-output system.

The four dc gains are calculated, sorted and plotted in the code below.

```
%       define radial and circumferential forces applied at four coil force nodes
%       "x" is radial, "y" is circumferential, total force is unity

        fcoil = 0.25;

        n24061fx = fcoil*sin(9.1148*pi/180);
        n24061fy = fcoil*cos(9.1148*pi/180);

        n24066fx = fcoil*sin(15.1657*pi/180);
        n24066fy = fcoil*cos(15.1657*pi/180);

        n24082fx = -fcoil*sin(15.1657*pi/180);
        n24082fy = fcoil*cos(15.1657*pi/180);

        n24087fx = -fcoil*sin(9.1148*pi/180);
        n24087fy = fcoil*cos(9.1148*pi/180);

%       define radial and circumferential forces applied at ends of piezo element
%       "x" is radial, "y" is circumferential, total force is unity

        fpz = 0.2/6;            % six nodes at each end of the piezo

%       bottom arm radial force

        n21538fx = fpz*cos(20.4549*pi/180);
        n21546fx = -fpz*cos(13.5298*pi/180);
        n21576fx = fpz*cos(20.4549*pi/180);
        n21584fx = -fpz*cos(13.5298*pi/180);
        n21617fx = fpz*cos(20.4549*pi/180);
        n21625fx = -fpz*cos(13.5298*pi/180);
        n22538fx = fpz*cos(20.4549*pi/180);
        n22546fx = -fpz*cos(13.5298*pi/180);
        n22576fx = fpz*cos(20.4549*pi/180);
        n22584fx = -fpz*cos(13.5298*pi/180);
        n22617fx = fpz*cos(20.4549*pi/180);
        n22625fx = -fpz*cos(13.5298*pi/180);

%       top arm radial force

        n24538fx = -fpz*cos(20.4549*pi/180);
        n24546fx = fpz*cos(13.5298*pi/180);
        n24576fx = -fpz*cos(20.4549*pi/180);
        n24584fx = fpz*cos(13.5298*pi/180);
        n24617fx = -fpz*cos(20.4549*pi/180);
        n24625fx = fpz*cos(13.5298*pi/180);
        n25538fx = -fpz*cos(20.4549*pi/180);
        n25546fx = fpz*cos(13.5298*pi/180);
        n25576fx = -fpz*cos(20.4549*pi/180);
```

```
n25584fx = fpz*cos(13.5298*pi/180);
n25617fx = -fpz*cos(20.4549*pi/180);
n25625fx = fpz*cos(13.5298*pi/180);

%       bottom arm circumferential force

n21538fy = -fpz*sin(20.4549*pi/180);
n21546fy = fpz*sin(13.5298*pi/180);
n21576fy = -fpz*sin(20.4549*pi/180);
n21584fy = fpz*sin(13.5298*pi/180);
n21617fy = -fpz*sin(20.4549*pi/180);
n21625fy = fpz*sin(13.5298*pi/180);
n22538fy = -fpz*sin(20.4549*pi/180);
n22546fy = fpz*sin(13.5298*pi/180);
n22576fy = -fpz*sin(20.4549*pi/180);
n22584fy = fpz*sin(13.5298*pi/180);
n22617fy = -fpz*sin(20.4549*pi/180);
n22625fy = fpz*sin(13.5298*pi/180);

%       top arm circumferential force

n24538fy = fpz*sin(20.4549*pi/180);
n24546fy = -fpz*sin(13.5298*pi/180);
n24576fy = fpz*sin(20.4549*pi/180);
n24584fy = -fpz*sin(13.5298*pi/180);
n24617fy = fpz*sin(20.4549*pi/180);
n24625fy = -fpz*sin(13.5298*pi/180);
n25538fy = fpz*sin(20.4549*pi/180);
n25546fy = -fpz*sin(13.5298*pi/180);
n25576fy = fpz*sin(20.4549*pi/180);
n25584fy = -fpz*sin(13.5298*pi/180);
n25617fy = fpz*sin(20.4549*pi/180);
n25625fy = -fpz*sin(13.5298*pi/180);

%       two-input system

%               first input is coil force
%               second input is excitation of both piezo elements with opposite polarity

%       f_coil is the vector of forces applied to coil

f_coil = [zeros(14,1)
          n24061fx
          n24066fx
          n24082fx
          n24087fx
          zeros(26,1)
          n24061fy
          n24066fy
          n24082fy
          n24087fy
          zeros(12,1)];

%       f_piezo is vector of forces applied to piezo ends
```

```
f_piezo = [     0
                0
                n21538fx  %       bottom arm radial force
                n21546fx
                n21576fx
                n21584fx
                n21617fx
                n21625fx
                n22538fx
                n22546fx
                n22576fx
                n22584fx
                n22617fx
                n22625fx
                0
                0
                0
                0
                n24538fx  %       top arm radial force
                n24546fx
                n24576fx
                n24584fx
                n24617fx
                n24625fx
                n25538fx
                n25546fx
                n25576fx
                n25584fx
                n25617fx
                n25625fx
                 0
                 0
                n21538fy  %       bottom arm circumferential force
                n21546fy
                n21576fy
                n21584fy
                n21617fy
                n21625fy
                n22538fy
                n22546fy
                n22576fy
                n22584fy
                n22617fy
                n22625fy
                 0
                 0
                 0
                 0
                n24538fy  %       top arm circumferential force
                n24546fy
                n24576fy
                n24584fy
                n24617fy
                n24625fy
                n25538fy
```

```
                                    n25546fy
                                    n25576fy
                                    n25584fy
                                    n25617fy
                                    n25625fy ];

%       define composite forcing function, force applied to each node times
%       eigenvector value for that node

        force_coil = f_coil'*xn;

        force_piezo = f_piezo'*xn;

%       prompt for uniform or variable zeta

        zeta_type = input('enter "1" to read in damping vector (zetain.m) …
                    or "enter" for uniform damping ...  ');

        if  (isempty(zeta_type))

                zeta_type = 0;

                zeta_uniform = input('enter value for uniform damping, …
                        .005 is 0.5% of critical (default) ... ');

                if (isempty(zeta_uniform))
                        zeta_uniform = 0.005;
                   end

                zeta_unsort = zeta_uniform*ones(num_modes_total,1);

        else

                zetain;      % read in zeta_unsort damping vector from zetain.m file

        end

        if  length(zeta_unsort) ~= num_modes_total

                error(['error - zetain vector has ',num2str(length(zeta_unsort)), …
                        ' entries instead of ',num2str(num_modes_total)]);

        end

%       define dc gains, 31 is head 1, 32 is head 0

        omega2 = (2*pi*freqvec)'.^2;     %  convert to radians and square

%       define frequency range for frequency response

        freqlo = 501;

        freqhi = 25000;

        flo=log10(freqlo) ;
```

```
        fhi=log10(freqhi) ;

        f=logspace(flo,fhi,300) ;
        frad=f*2*pi ;

%       calculate dc gains if uniform damping, peak gains if non-uniform

        if  zeta_type == 0                  % dc gain

                gain_h0_coil = abs([force_coil(1)*xn(32,1)/frad(1) ...
                force_coil(2:num_modes_total).*xn(32,2:num_modes_total) ...
                        ./omega2(2:num_modes_total)]);

                gain_h1_coil = abs([force_coil(1)*xn(31,1)/frad(1) ...
                force_coil(2:num_modes_total).*xn(31,2:num_modes_total) ...
                ./omega2(2:num_modes_total)]);

                gain_h0_piezo = abs([force_piezo(1)*xn(31,1)/frad(1) ...
                force_piezo(2:num_modes_total).*xn(32,2:num_modes_total) …
                ./omega2(2:num_modes_total)]);

                gain_h1_piezo = abs([force_piezo(1)*xn(31,1)/frad(1) ...
                force_piezo(2:num_modes_total).*xn(31,2:num_modes_total) ...
                ./omega2(2:num_modes_total)]);

        elseif  zeta_type == 1              % peak gain

                gain_h0_coil = abs([force_coil(1)*xn(32,1)/frad(1) ...
                force_coil(2:num_modes_total).*xn(32,2:num_modes_total) ...
                ./((2*zeta_unsort(2:num_modes_total))'.*omega2(2:num_modes_total))]);

                gain_h1_coil = abs([force_coil(1)*xn(31,1)/frad(1) ...
                force_coil(2:num_modes_total).*xn(31,2:num_modes_total) ...
                ./((2*zeta_unsort(2:num_modes_total))'.*omega2(2:num_modes_total))]);

                gain_h0_piezo = abs([force_piezo(1)*xn(31,1)/frad(1) ...
                force_piezo(2:num_modes_total).*xn(32,2:num_modes_total) ...
                ./((2*zeta_unsort(2:num_modes_total))'.*omega2(2:num_modes_total))]);

                gain_h1_piezo = abs([force_piezo(1)*xn(31,1)/frad(1) ...
                force_piezo(2:num_modes_total).*xn(31,2:num_modes_total) ...
                ./((2*zeta_unsort(2:num_modes_total))'.*omega2(2:num_modes_total))]);

        end

%       sort gains, keeping track of original and new indices so can rearrange
%       eigenvalues and eigenvectors

        [gain_h0_coil_sort,index_h0_coil_sort] = sort(gain_h0_coil);

        [gain_h1_coil_sort,index_h1_coil_sort] = sort(gain_h1_coil);

        [gain_h0_piezo_sort,index_h0_piezo_sort] = sort(gain_h0_piezo);

        [gain_h1_piezo_sort,index_h1_piezo_sort] = sort(gain_h1_piezo);
```

```
gain_h0_coil_sort = fliplr(gain_h0_coil_sort);          % max to min

gain_h1_coil_sort = fliplr(gain_h1_coil_sort);          % max to min

gain_h0_piezo_sort = fliplr(gain_h0_piezo_sort);        % max to min

gain_h1_piezo_sort = fliplr(gain_h1_piezo_sort);        % max to min

index_h0_coil_sort = fliplr(index_h0_coil_sort)         % max to min indices

index_h1_coil_sort = fliplr(index_h1_coil_sort)         % max to min indices

index_h0_piez_sort = fliplr(index_h0_piezo_sort)        % max to min indices

index_h1_piez_sort = fliplr(index_h1_piezo_sort)        % max to min indices

index_orig = 1:num_modes_total;

[index_h0_coil_sort' index_h1_coil_sort' index_h0_piez_sort' index_h1_piez_sort']

%       plot results

        semilogy(index_orig(2:num_modes_total),freqvec(2:num_modes_total),'k-');
        title(['frequency versus mode number'])
        xlabel('mode number')
        ylabel('frequency, hz')
        grid off
        disp('execution paused to display figure, "enter" to continue');%pause

        semilogy(index_orig,gain_h0_coil,'k.-',index_orig,gain_h1_coil,'k-')
        title(['coil input:  dc value of each mode contribution versus mode number'])
        xlabel('mode number')
        ylabel('dc value')
        legend('h0 coil input','h1 coil input')
        grid off
        disp('execution paused to display figure, "enter" to continue');%pause

        semilogy(index_orig,gain_h0_piezo,'k.-',index_orig,gain_h1_piezo,'k-')
        title(['piezo input:  dc value of each mode contribution versus mode number'])
        xlabel('mode number')
        ylabel('dc value')
        legend('h0 piezo input','h1 piezo input')
        grid off
        disp('execution paused to display figure, "enter" to continue');%pause

        loglog(freqvec(2:num_modes_total),gain_h0_coil(2:num_modes_total),'k.-', ...
                freqvec(2:num_modes_total),gain_h1_coil(2:num_modes_total),'k-')
        title(['coil input:  dc value of each mode contribution versus frequency'])
        xlabel('frequency, hz')
        ylabel('dc value')
        axis([500 25000 -inf inf])
        legend('h0 coil input','h1 coil input')
        grid off
        disp('execution paused to display figure, "enter" to continue');%pause
```

```
loglog(freqvec(2:num_modes_total),gain_h0_piezo(2:num_modes_total),'k.-', ...
                freqvec(2:num_modes_total),gain_h1_piezo(2:num_modes_total),'k-')
title(['piezo input:  dc value of each mode contribution versus frequency'])
xlabel('frequency, hz')
ylabel('dc value')
axis([500 25000 -inf inf])
legend('h0 piezo input','h1 piezo input')
grid off
disp('execution paused to display figure, "enter" to continue');%pause

semilogy(index_orig,gain_h0_coil_sort,'k.-',index_orig,gain_h1_coil_sort,'k-')
title(['coil input:  sorted dc value of each mode versus number of modes included'])
xlabel('modes included')
ylabel('sorted dc value')
legend('h0 coil input','h1 coil input')
grid off
disp('execution paused to display figure, "enter" to continue');%pause

semilogy(index_orig,gain_h0_piezo_sort,'k.-',index_orig,gain_h1_piezo_sort,'k-')
title(['piezo input:  sorted dc value of each mode versus number of modes included'])
xlabel('modes included')
ylabel('sorted dc value')
legend('h0 piezo input','h1 piezo input')
grid off
disp('execution paused to display figure, "enter" to continue');%pause

semilogy(index_orig,gain_h0_coil_sort,'k.-',index_orig,gain_h1_coil_sort,'k.-', ...
                        index_orig,gain_h0_piezo_sort, …
                       'k-',index_orig,gain_h1_piezo_sort,'k-')
title(['coil and piezo input:  sorted dc value of each mode versus number …
                        of modes included'])
xlabel('modes included')
ylabel('sorted dc value')
legend('h0 coil input','h1 coil input','h0 piezo input','h1 piezo input')
grid off
disp('execution paused to display figure, "enter" to continue');%pause
```

Figure 19.22 repeats Figure 19.5, plotting resonant frequency versus mode number. Note that there are several "jumps" in the curve, the most significant between mode 4 and mode 5. As indicated in Section 17.6, "jumps" in the frequency plot can indicate the system transitioning from one type of characteristic motion to another. In this case modes 2, 3 and 4 involve bending motions of the system, while mode 5 involves coil torsion.

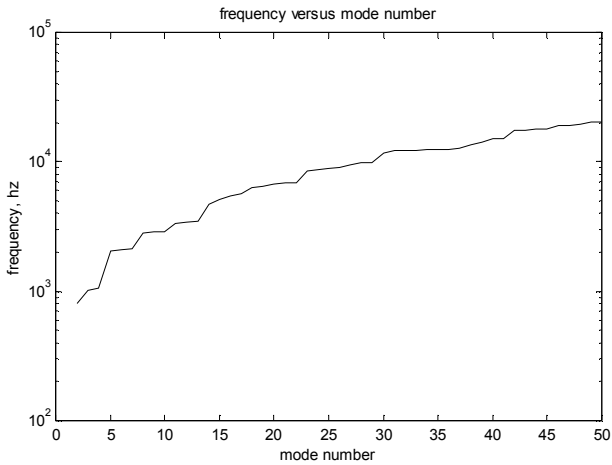

Figure 19.22: Resonant frequencies versus mode number.

The dc gains for head 0 and head 1 for coil input are shown in Figure 19.23. Because the actuator is nearly symmetrical in design the gains of the two heads are quite similar.

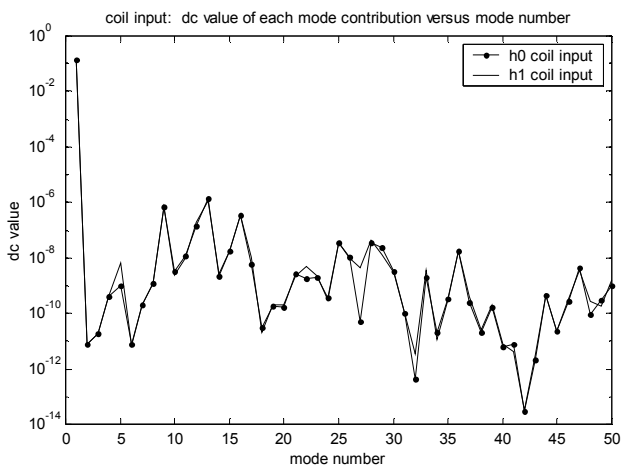

Figure 19.23: dc gain versus mode for both heads for coil input.

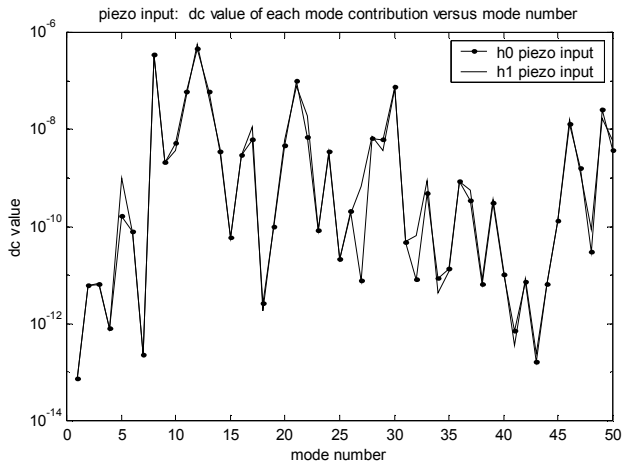

Figure 19.24: dc gain versus mode number for both heads for piezo input.

The gains for both heads for piezo inputs are shown in Figure 19.24.

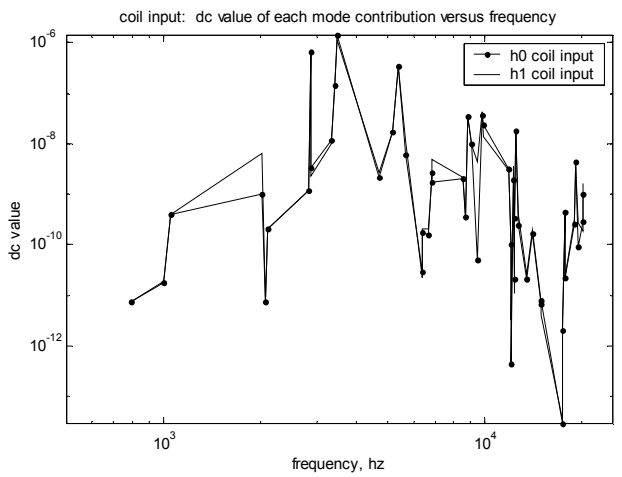

Figure 19.25: dc gain versus frequency for both heads for coil input.

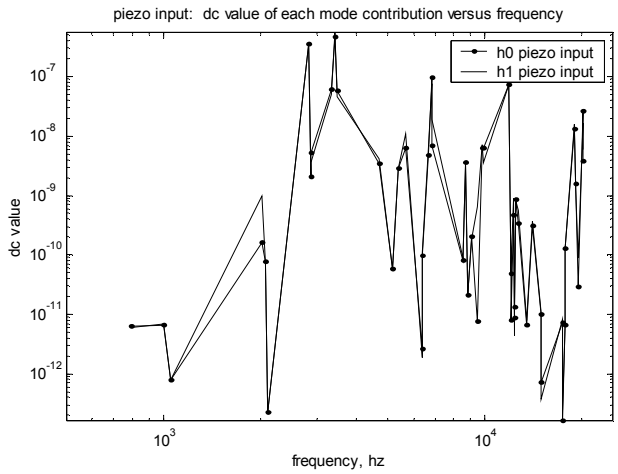

Figure 19.26: dc gain versus frequency for both heads for piezo input.

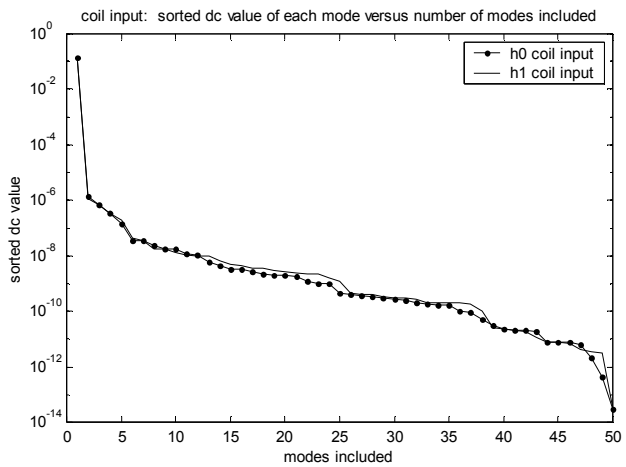

Figure 19.27: Sorted dc gain for both heads for coil input.

The sorted dc gains of the two heads, Figure 19.27, are very similar because the actuator design is so symmetrical.

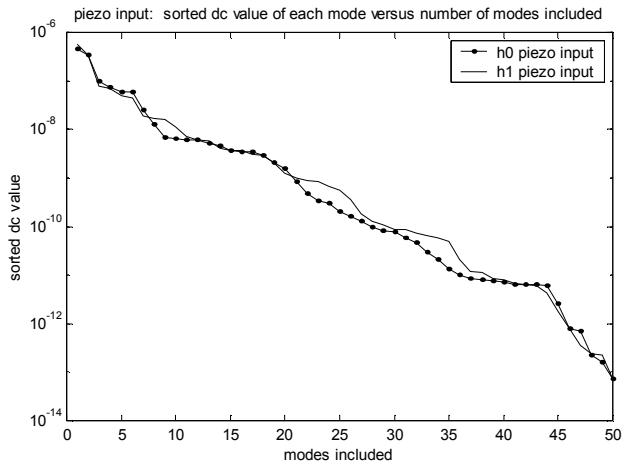

Figure 19.28: Sorted dc gain for both heads for piezo input.

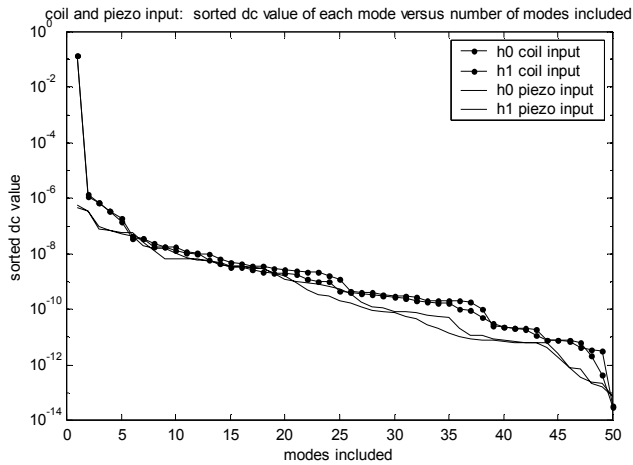

Figure 19.29: Sorted dc gain for both heads for both coil and piezo inputs.

The sorted gains of head 0 and head 1 for both coil and piezo inputs can be seen in Figure 19.29. They are of similar magnitude because the piezo force "fpz" in Section 19.5.2 was chosen to be 0.2.

With the partial listing of mode ranking for both heads and both inputs shown in Table 19.1, we can start looking at the difficulties of using dc and peak gains for ranking MIMO systems.

Table 19.1 lists the mode ranking for the first 15 modes for:

Column 1: head 0, coil input

Column 2: head 1, coil input

Column 3: head 0, piezo input

Column 4: head 1, piezo input

1	1	12	12
13	13	8	8
9	9	21	21
16	16	30	30
12	12	11	11
28	28	13	13
25	25	49	22
29	36	46	49
36	15	22	46
15	29	28	17
11	17	17	28
26	11	29	20
17	26	10	50
47	5	20	14
10	22	50	29

Table 19.1: Ranking for first 15 modes for head 0 and head 1 for coil and piezo inputs.

The first two columns in Table 19.1 show that for coil input, head 0 and head 1 have the same ranking through the first seven modes, then their rankings change. The second two columns show that for piezo input, head 0 and head 1 have the same ranking through the first six modes, then their rankings change.

If one were to choose a single ranking for the model which would take into account both inputs and both outputs, it is difficult to see how to do it given the rankings in the table. Thus the necessity of balanced reduction for MIMO models. (See Problem P19.1 for using dc gain to rank for reduction.)

19.5.3 Building State Space Matrices

In this section of code the system matrices are assembled and the four frequency responses are plotted. For all previous SISO models in the book we have built the system matrices using dc gain ordering of modes. Here, for the MIMO model, we will assemble the system using the original, unsorted ordering and will let "balreal" do all the work of sorting in the next section.

```
%       create five state space systems with all modes included, differing in the ordering
%       of the modes, the unsorted system will be used for all reductions, letting balreal do all
```

```
%       the ordering, the sorted systems will be used to show how the dc gain ordering
%       compares with the balanced ordering

%               1)  unsorted
%               2)  sorted, head 0, coil input
%               3)  sorted, head 1, coil input
%               4)  sorted, head 0, piezo input
%               5)  sorted, head 1, piezo input

        for  num_model = 1:5

        if  num_model == 1                      % unsorted

                xnnew = xn;

                freqnew = freqvec;

        elseif  num_model == 2                  % sorted, head 0, coil input

                xnnew = xn(:,index_h0_coil_sort);

                freqnew = freqvec(index_h0_coil_sort);

        elseif  num_model == 3                  % sorted, head 1, coil input

                xnnew = xn(:,index_h1_coil_sort);

                freqnew = freqvec(index_h1_coil_sort);

        elseif  num_model == 4                  % sorted, head 0, piezo input

                xnnew = xn(:,index_h0_piezo_sort);

                freqnew = freqvec(index_h0_piezo_sort);

        elseif  num_model == 5                  % sorted, head 1, piezo input

                xnnew = xn(:,index_h1_piezo_sort);

                freqnew = freqvec(index_h1_piezo_sort);

        end

%       define variables for all modes included system matrix, a

        w = freqnew*2*pi;               %       frequencies in rad/sec

        w2 = w.^2;

       zw = 2*zeta_unsort.*w;

%       define size of system matrix

        asize = 2*num_modes_total;
```

```
        disp('   ');
        disp('   ');
        disp(['size of system matrix a is ',num2str(asize)]);

%       setup system matrix for all modes included model

        a = zeros(asize);

        for  col = 2:2:asize

        row = col-1;

        a(row,col) = 1;

        end

        for  col = 1:2:asize

        row = col+1;

        a(row,col) = -w2((col+1)/2);

        end

        for  col = 2:2:asize

        row = col;

        a(row,col) = -zw(col/2);

        end

%       setup input matrix b, state space forcing function in principal coordinates

%       two-input system

%               first input is coil force
%               second input is excitation of both piezo elements with opposite polarity

        f_physical = [f_coil f_piezo];

%       f_principal is the matrix of forces in principal coordinates

        f_principal = xnnew'*f_physical;

%       b is the matrix of forces in principal coordinates, state space form

        b = zeros(2*num_modes_total,2);

        for  cnt = 1:num_modes_total

                b(2*cnt,:) = f_principal(cnt,:);

        end
```

```
%       setup cdisp and cvel, padded xn matrices to give the displacement and velocity
%       vectors in physical coordinates cdisp and cvel each have numdof rows
%       and alternating columns consisting of columns of xnnew and zeros to give total
%       columns equal to the number of states

%       all modes included cdisp and cvel

        for  col = 1:2:2*length(freqnew)

        for  row = 1:numdof

        c_disp(row,col) = xnnew(row,ceil(col/2));

        cvel(row,col) = 0;

        end

        end

        for  col = 2:2:2*length(freqnew)

        for  row = 1:numdof

        c_disp(row,col) = 0;

              cvel(row,col) = xnnew(row,col/2);

        end

        end

%       define output

        d = [0];  %

        if  num_model == 1                          % unsorted

                    sys = ss(a,b,c_disp(31:32,:),d);

        elseif  num_model == 2                      % sorted, head 0, coil input

                    sys_h0_coil = ss(a,b,c_disp(31:32,:),d);

        elseif  num_model == 3                      % sorted, head 1, coil input

                    sys_h1_coil = ss(a,b,c_disp(31:32,:),d);

        elseif  num_model == 4                      % sorted, head 0, piezo input

                    sys_h0_piezo = ss(a,b,c_disp(31:32,:),d);

        elseif  num_model == 5                      % sorted, head 1, piezo input

                    sys_h1_piezo = ss(a,b,c_disp(31:32,:),d);
```

```
        end

        end                 % end of for loop for creating system matrices
```

19.5.4 Balancing, Reduction

Balancing the system involves calculating gramians, which are only defined for negative definite systems. This requires separating the rigid body mode from the oscillatory modes and balancing the oscillatory modes. The system matrices are partitioned and a model of only oscillatory modes is created and balanced. Plotting the diagonal gramian terms (squares of the Hankel singular values) reveals the relative importance of the states.

Modred is used to reduce the states using both the "del" and "mdc" options. The complete system is rebuilt by augmenting the rigid body mode (states) with the reduced oscillatory modes (states). Frequency responses are then plotted, comparing the two reducing methods with the original 50-mode model.

```
%       partition system matrices into rigid body mode and oscillatory modes, can't use balreal
%       with rigid body mode so will reduce the oscillatory modes and then augment the
%       resulting system with the rigid body mode

%       define oscillatory system, where output 31 is head 1, output 32 is head 0

        [a,b,c_disp,d] = ssdata(sys);

        a_syso = a(3:asize,3:asize);

        b_syso = b(3:asize,:);

        c_syso = c_disp(1:2,3:asize);

        syso = ss(a_syso,b_syso,c_syso,d);

%       define controllability and observability gramians for oscillatory system, syso

        wc = gram(syso,'c');

        wo = gram(syso,'o');

        [row_syso,col_syso] = size(a_syso);

        statevec = 1:row_syso;

%       plot controllability and observability gramians

        meshz(wc);
        view(60,30);
```

```
title(['controllability gramian for oscillatory system'])
xlabel('state')
ylabel('state')
grid on

disp('execution paused to display figure, "enter" to continue');%pause

meshz(wo);
view(60,30);
title(['observability gramian for oscillatory system'])
xlabel('state')
ylabel('state')
grid on

disp('execution paused to display figure, "enter" to continue');%pause

%       pull out diagonal elements

wc_diag = diag(wc);

wo_diag = diag(wo);

%       plot diagonal terms of controllability and observability gramians

semilogy(statevec,wc_diag,'k.-')
title(['controllability gramian diagonal terms'])
xlabel('states')
ylabel('diagonal')
grid off

disp('execution paused to display figure, "enter" to continue');%pause

semilogy(statevec,wo_diag,'k.-')
title(['observability gramian diagonal terms'])
xlabel('states')
ylabel('diagonal')
grid off

disp('execution paused to display figure, "enter" to continue');%pause

%       position and velocity states plotted separately

semilogy(statevec(1:2:row_syso),wc_diag(1:2:row_syso),'k.-', ...

statevec(2:2:row_syso),wc_diag(2:2:row_syso),'k-')
title(['controllability gramian diagonal terms'])
xlabel('states')
ylabel('diagonal')
legend('position states','velocity states',3)
grid off

disp('execution paused to display figure, "enter" to continue');%pause

semilogy(statevec(1:2:row_syso),wo_diag(1:2:row_syso),'k.-', ...
```

```
statevec(2:2:row_syso),wo_diag(2:2:row_syso),'k-')
title(['observability gramian diagonal terms'])
xlabel('states')
ylabel('diagonal')
legend('position states','velocity states',3)
grid off

disp('execution paused to display figure, "enter" to continue');%pause

%       use balreal to rank oscillatory states and modred to reduce for comparison

[sysob,g,T,Ti] = balreal(syso);

[ao_bal,bo_bal,cdispo_bal,do_bal] = ssdata(sysob);

semilogy(g,'k.-')
title('diagonal of balanced gramian versus number of states')
xlabel('state number')
ylabel('diagonal of balanced gramian')
grid off

osc_states_used = input(['enter number of oscillatory states to use, default 20  ...  ']);

if  isempty(osc_states_used)

        osc_states_used = 20;

end

num_modes_used = 1 + osc_states_used/2;      % number of modes for overlaid plots

%       use modred to order oscillatory states from balreal to define reduced order
%       oscillatory system using both "del" and "mdc"

rsys_delo = modred(sysob,osc_states_used+1:2*num_modes_total-2,'del');

rsys_mdco = modred(sysob,osc_states_used+1:2*num_modes_total-2,'mdc');

%       rebuild system by appending balanced realization of oscillatory modes to rigid
%       body mode

[a_delo_bal,b_delo_bal,c_delo_bal,d_delo_bal] = ssdata(rsys_delo);

a_del_bal = [   a(1:2,1:2)      zeros(2,osc_states_used)
                zeros(osc_states_used,2)      a_delo_bal    ];

b_del_bal = [b(1:2,:)
                b_delo_bal];

c_del_bal = [c_disp(1:2,1:2) c_delo_bal];

rsys_del = ss(a_del_bal,b_del_bal,c_del_bal,d);

[a_mdco_bal,b_mdco_bal,c_mdco_bal,d_mdco_bal] = ssdata(rsys_mdco);
```

```
        a_mdc_bal = [   a(1:2,1:2)      zeros(2,osc_states_used)
                                        zeros(osc_states_used,2)      a_mdco_bal    ];

        b_mdc_bal = [b(1:2,:)
                                        b_mdco_bal];

        c_mdc_bal = [c_disp(1:2,1:2) c_mdco_bal];

        rsys_mdc = ss(a_mdc_bal,b_mdc_bal,c_mdc_bal,d);

%       frequency response for unsorted system

        [mag,phs] = bode(sys,frad);

%       plot original system response, output of bode command has dimensions
%       of "i" x "j" x "k" where "i" is output row, "j" is input column and "k" is the
%       vector of frequencies

        magh0coil = mag(2,1,:);
        magh1coil = mag(1,1,:);
        magh0pz   = mag(2,2,:);
        magh1pz   = mag(1,2,:);

        loglog(f,magh0coil(1,:),'k.-',f,magh1coil(1,:),'k-')
        title(['gap displacement, all ',num2str(num_modes_total),' modes included'])
        xlabel('Frequency, hz')
        ylabel('Magnitude, mm')
        axis([500 25000 1e-9 2e-4])
        legend('head 0, coil input','head 1, coil input',3)
        grid off
        disp('execution paused to display figure, "enter" to continue');%pause

        loglog(f,magh0pz(1,:),'k.-',f,magh1pz(1,:),'k-')
        title(['gap displacement, all ',num2str(num_modes_total),' modes included'])
        xlabel('Frequency, hz')
        ylabel('Magnitude, mm')
        axis([500 25000 1e-9 2e-4])
        legend('head 0 piezo input','head 1 piezo input',3)
        grid off
        disp('execution paused to display figure, "enter" to continue');%pause

        loglog(f,magh0coil(1,:),'k.-',f,magh1coil(1,:),'k.-',f,magh0pz(1,:),'k-',f,magh1pz(1,:),'k-
')
        title(['gap displacement, all ',num2str(num_modes_total),' modes included'])
        xlabel('Frequency, hz')
        ylabel('Magnitude, mm')
        axis([500 25000 1e-9 2e-4])
        legend('head 0, coil input','head 1, coil input','head 0 piezo input','head 1 piezo …
                                        input',3)
        grid off
        disp('execution paused to display figure, "enter" to continue');%pause

%       frequency response for balanced reduced modred "del"
```

```
[magr_del,phsr_del] = bode(rsys_del,frad);

magr_delh0coil = magr_del(2,1,:);
magr_delh1coil = magr_del(1,1,:);
magr_delh0pz   = magr_del(2,2,:);
magr_delh1pz   = magr_del(1,2,:);

loglog(f,magr_delh0coil(1,:),'k-',f,magr_delh1coil(1,:),'k.-',f,magr_delh0pz(1,:), …
                        'k.-',f,magr_delh1pz(1,:),'k-')
title(['gap displacement, modred "del", ',num2str(osc_states_used), …
                ' oscillatory states included'])
xlabel('Frequency, hz')
ylabel('Magnitude, mm')
axis([500 25000 1e-9 2e-4])
legend('head 0, coil input','head 1, coil input','head 0 piezo input' …
                ,'head 1 piezo input',3)
grid off
disp('execution paused to display figure, "enter" to continue');%pause

loglog(f,magh0coil(1,:),'k-',f,magr_delh0coil(1,:),'k.-')
title(['gap displacement, modred "del", ',num2str(osc_states_used), …
                ' oscillatory states included'])
xlabel('Frequency, hz')
ylabel('Magnitude, mm')
axis([500 25000 1e-9 2e-4])
legend('head 0, coil input','"del" reduced head 0, coil input',3)
grid off
disp('execution paused to display figure, "enter" to continue'); pause

loglog(f,magh1coil(1,:),'k-',f,magr_delh1coil(1,:),'k.-')
title(['gap displacement, modred "del", ',num2str(osc_states_used), …
                ' oscillatory states included'])
xlabel('Frequency, hz')
ylabel('Magnitude, mm')
axis([500 25000 1e-9 2e-4])
legend('head 1, coil input','"del" reduced head 1, coil input',3)
grid off
disp('execution paused to display figure, "enter" to continue');%pause

loglog(f,magh0pz(1,:),'k-',f,magr_delh0pz(1,:),'k.-')
title(['gap displacement, modred "del", ',num2str(osc_states_used), …
                ' oscillatory states included'])
xlabel('Frequency, hz')
ylabel('Magnitude, mm')
axis([500 25000 1e-9 2e-4])
legend('head 0, piezo input','"del" reduced head 0, piezo input',3)
grid off
disp('execution paused to display figure, "enter" to continue'); pause

loglog(f,magh1pz(1,:),'k-',f,magr_delh1pz(1,:),'k.-')
title(['gap displacement, modred "del", ',num2str(osc_states_used), …
                ' oscillatory states included'])
xlabel('Frequency, hz')
ylabel('Magnitude, mm')
axis([500 25000 1e-9 2e-4])
```

```
        legend('head 1, piezo input','"del" reduced head 1, piezo input',3)
        grid off
        disp('execution paused to display figure, "enter" to continue');%pause

%       frequency response for balanced reduced modred "mdc"

        [magr_mdc,phsr_mdc] = bode(rsys_mdc,frad);

        magr_mdch0coil = magr_mdc(2,1,:);
        magr_mdch1coil = magr_mdc(1,1,:);
        magr_mdch0pz   = magr_mdc(2,2,:);
        magr_mdch1pz   = magr_mdc(1,2,:);

        loglog(f,magr_mdch0coil(1,:),'k-',f,magr_mdch1coil(1,:),'k.-', …
                f,magr_mdch0pz(1,:),'k.-',f,magr_mdch1pz(1,:),'k-')
        title(['gap displacement, modred "mdc", ',num2str(osc_states_used), …
                ' oscillatory states included'])
        xlabel('Frequency, hz')
        ylabel('Magnitude, mm')
        axis([500 25000 1e-9 2e-4])
        legend('head 0, coil input','head 1, coil input','head 0 piezo input','head 1 piezo …
                input',3)
        grid off
        disp('execution paused to display figure, "enter" to continue');%pause

        loglog(f,magh0coil(1,:),'k-',f,magr_mdch0coil(1,:),'k.-')
        title(['gap displacement, modred "mdc", ',num2str(osc_states_used), …
                ' oscillatory states included'])
        xlabel('Frequency, hz')
        ylabel('Magnitude, mm')
        axis([500 25000 1e-9 2e-4])
        legend('head 0, coil input','"mdc" reduced head 0, coil input',3)
        grid off
        disp('execution paused to display figure, "enter" to continue'); pause

        loglog(f,magh1coil(1,:),'k-',f,magr_mdch1coil(1,:),'k.-')
        title(['gap displacement, modred "mdc", ',num2str(osc_states_used), …
                ' oscillatory states included'])
        xlabel('Frequency, hz')
        ylabel('Magnitude, mm')
        axis([500 25000 1e-9 2e-4])
        legend('head 1, coil input','"mdc" reduced head 1, coil input',3)
        grid off
        disp('execution paused to display figure, "enter" to continue');%pause

        loglog(f,magh0pz(1,:),'k-',f,magr_mdch0pz(1,:),'k.-')
        title(['gap displacement, modred "mdc", ',num2str(osc_states_used), …
                ' oscillatory states included'])
        xlabel('Frequency, hz')
        ylabel('Magnitude, mm')
        axis([500 25000 1e-9 2e-4])
        legend('head 0, piezo input','"mdc" reduced head 0, piezo input',3)
        grid off
        disp('execution paused to display figure, "enter" to continue'); pause
```

```
loglog(f,magh1pz(1,:),'k-',f,magr_mdch1pz(1,:),'k.-')
title(['gap displacement, modred "mdc", ',num2str(osc_states_used), …
                ' oscillatory states included'])
xlabel('Frequency, hz')
ylabel('Magnitude, mm')
axis([500 25000 1e-9 2e-4])
legend('head 1, piezo input','"mdc" reduced head 1, piezo input',3)
grid off
disp('execution paused to display figure, "enter" to continue');%pause
```

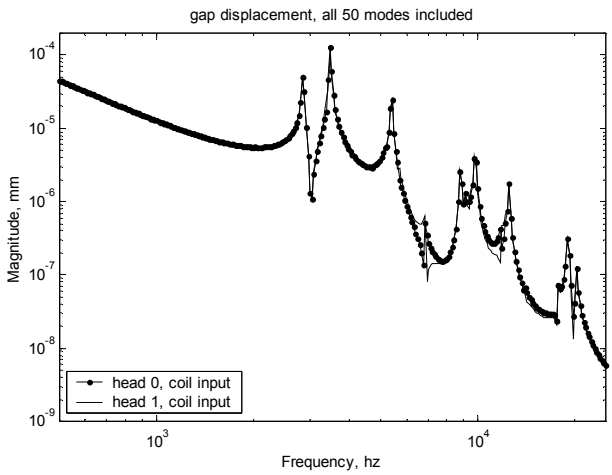

Figure 19.30: Frequency response for coil input for both heads, all modes included.

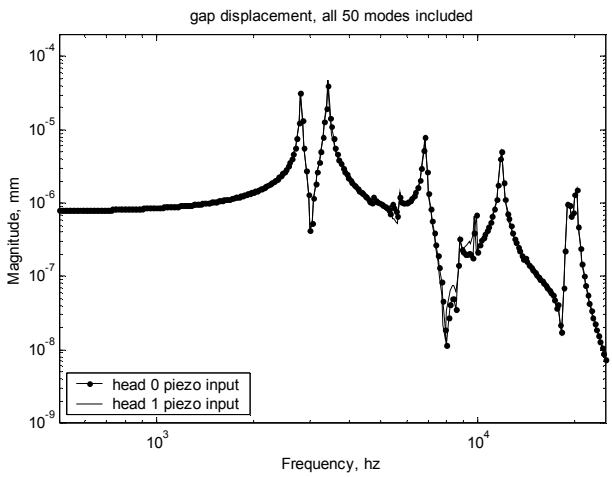

Figure 19.31: Frequency response for piezo input for both heads, all modes included.

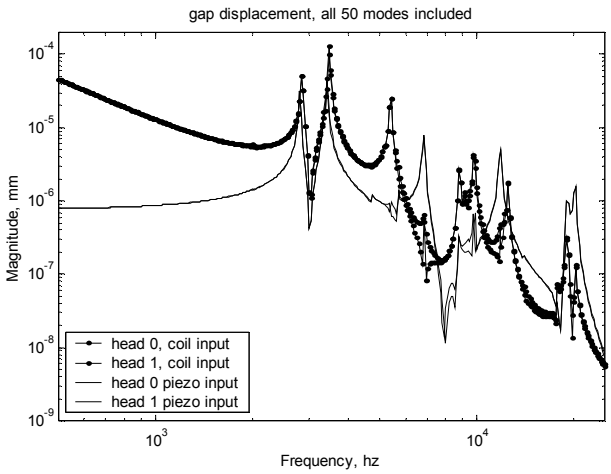

Figure 19.32: Frequency response for both coil and piezo inputs for both heads, all modes included.

The frequency response plots for both inputs and both outputs are shown above for reference.

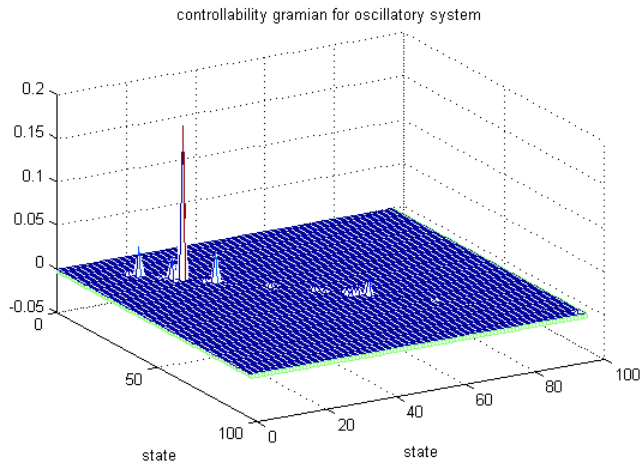

Figure 19.33: Controllability gramian values.

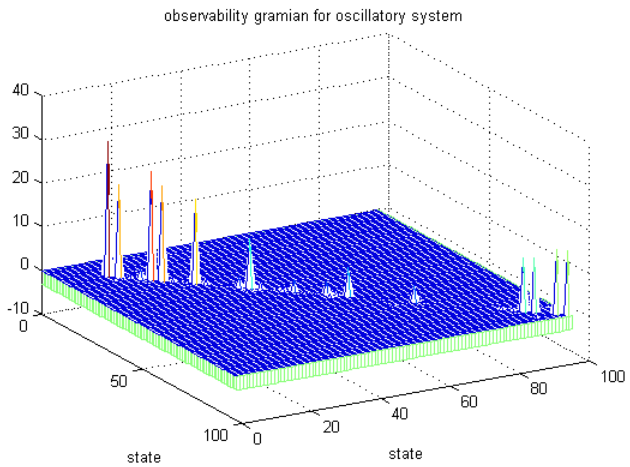

Figure 19.34: Observability gramian values.

Graphically, Figures 19.33 and 19.34 show the two gramians for this MIMO system. The gramians are nearly diagonal. The controllability gramian displays a predominance of lower frequency states, while the observability gramian has some higher frequency states included.

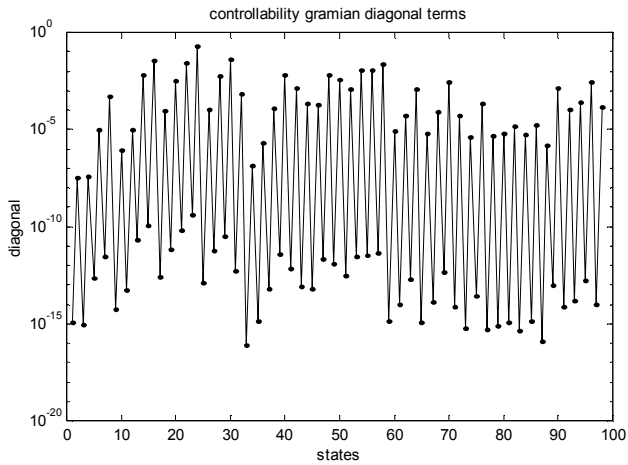

Figure 19.35: Controllability gramian diagonal terms versus states.

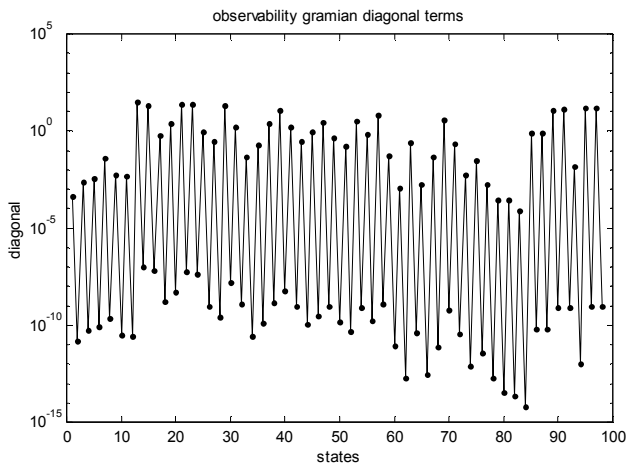

Figure 19.36: Observability gramian diagonal terms versus states.

Plotting the diagonal elements of the two gramians reveals the same pattern as for the SISO model. The maximum and minimum values for each mode are related by the square of the eigenvalue for that mode.

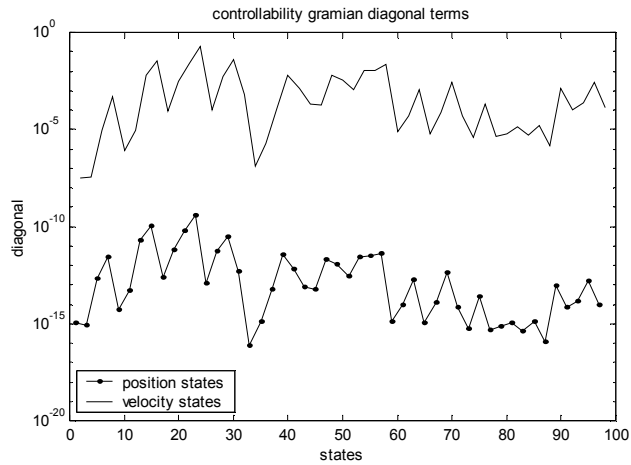

Figure 19.37: Controllability gramian diagonal position and velocity state terms.

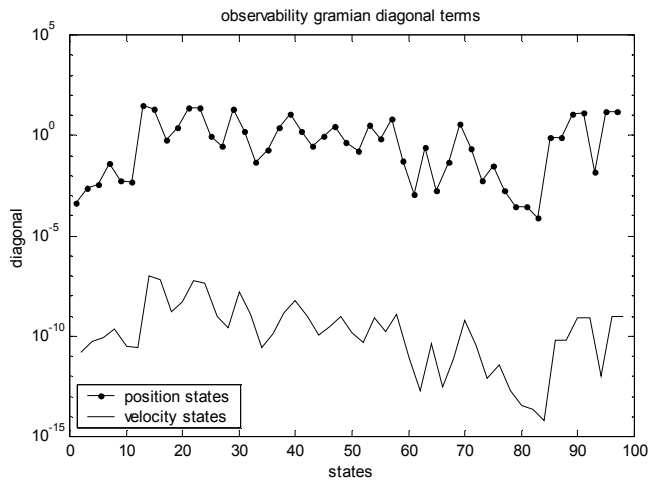

Figure 19.38: Observability gramian diagonal position and velocity state terms.

Plotting the position and velocity terms for each gramian separately displays their character on a mode-by-mode basis.

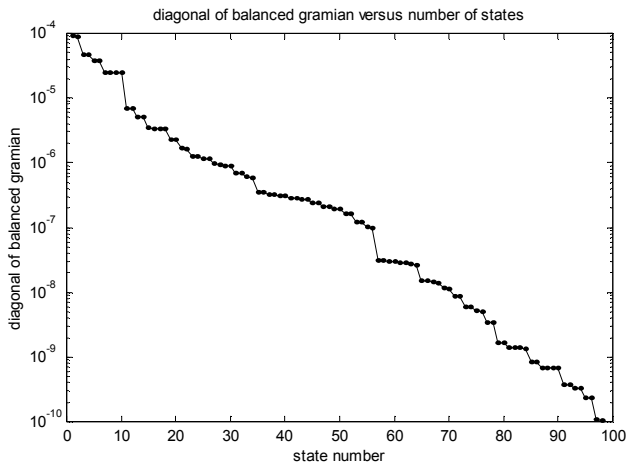

Figure 19.39: Balanced gramian diagonal terms (Hankel singular values) versus state number.

The balanced gramian shows several sharp drops in magnitude, one at 10 states and one at 56 states. We will see in Section 19.5.7 that 10 oscillatory modes (20 oscillatory states) are required for a normalized reduction index of less than 5% for coil input, and that 16 oscillatory modes (32 oscillatory states) are required for a normalized reduction index of less than 5% for piezo input.

19.5.5 Frequency Responses for Different Numbers of Retained States

This section displays pairs of frequency responses, one for head 0 for coil input and one for head 0 for piezo input. Each pair of plots represents an increasing number of oscillatory modes included in the reduced model. The original 50 mode model is overlaid to show the error in the reduced model. Note how the balanced method adds modes and which modes it chooses.

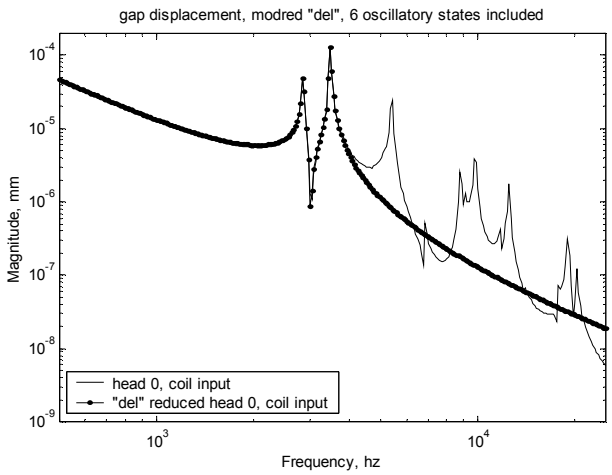

Figure 19.40: Head 0, coil input, six reduced oscillatory states included.

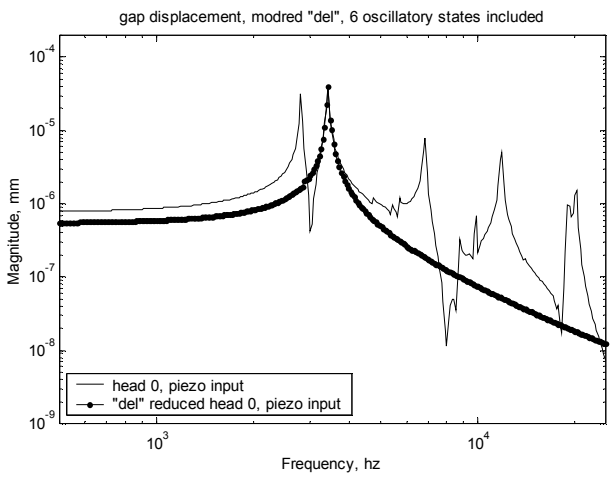

Figure 19.41: Head 0, piezo input, six reduced oscillatory states included.

With only six oscillatory states included the coil input captures the first two resonances but the piezo input misses the first resonance.

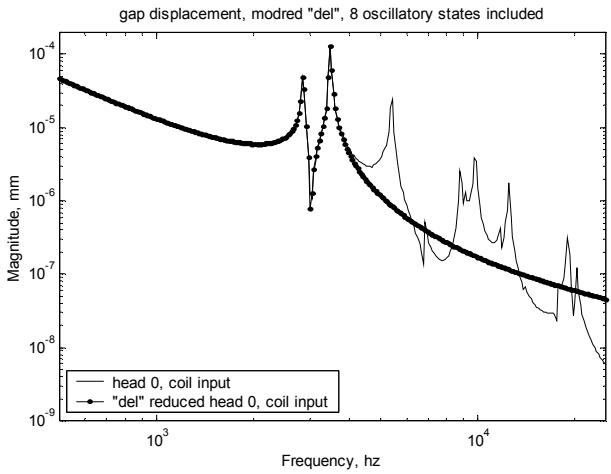

Figure 19.42: Head 0, coil input, eight reduced oscillatory states included.

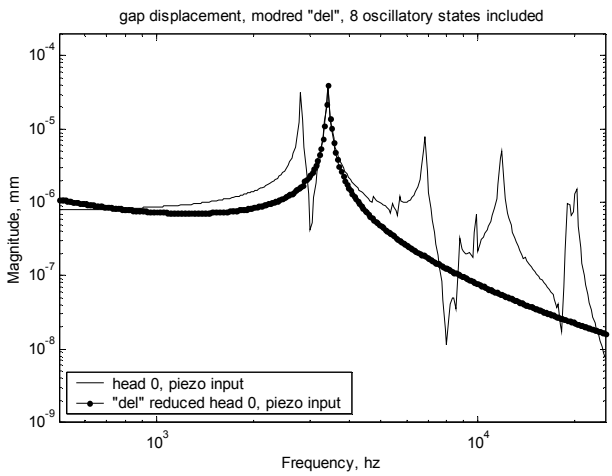

Figure 19.43: Head 0, piezo input, eight reduced oscillatory states included.

With 8 oscillatory states included the coil input captures the first two resonances but the piezo input again misses the first resonance.

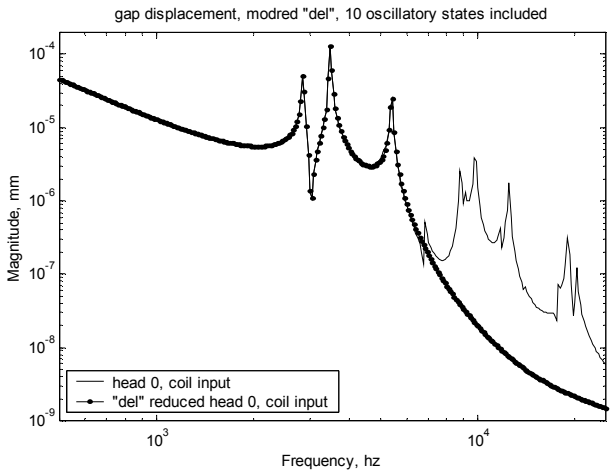

Figure 19.44: Head 0, coil input, 10 reduced oscillatory states included.

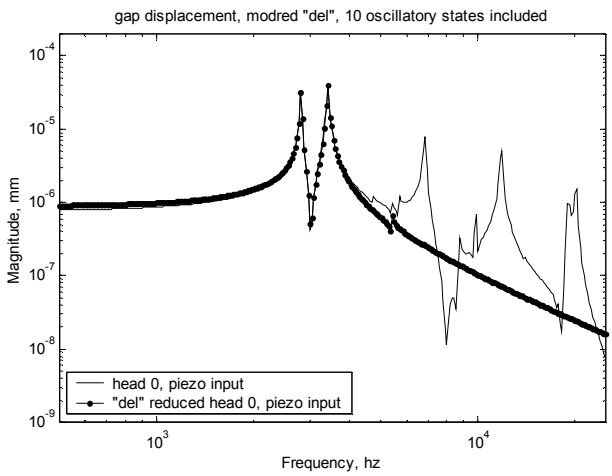

Figure 19.45: Head 0, piezo input, 10 reduced oscillatory states included.

With 10 oscillatory states included the first three coil input modes are fit well and also the first two piezo input modes.

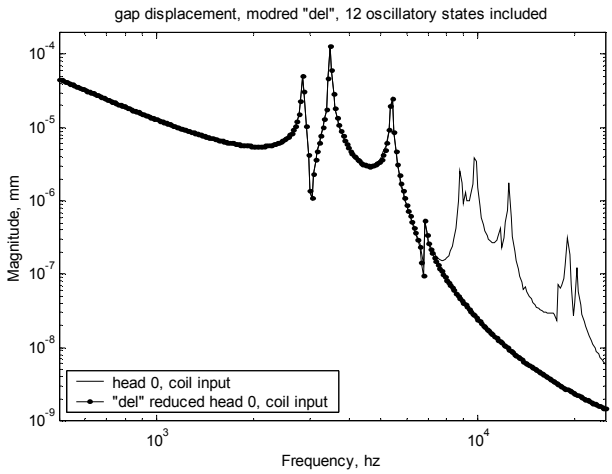

Figure 19.46: Head 0, coil input, 12 reduced oscillatory states included.

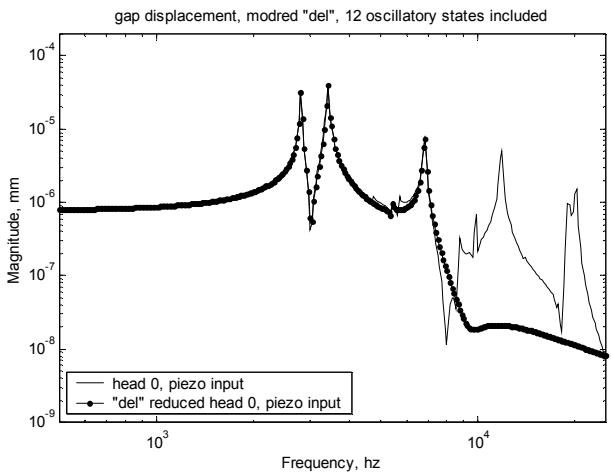

Figure 19.47: Head 0, piezo input, 12 reduced oscillatory states included.

With 12 oscillatory states included the first three major modes are fitted for both coil and piezo inputs.

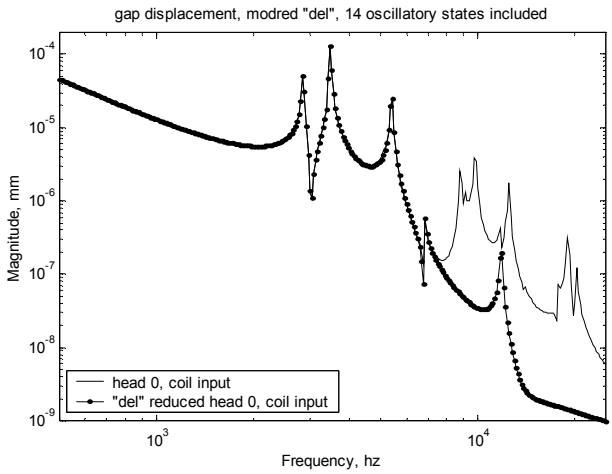

Figure 19.48: Head 0, coil input, 14 reduced oscillatory states included.

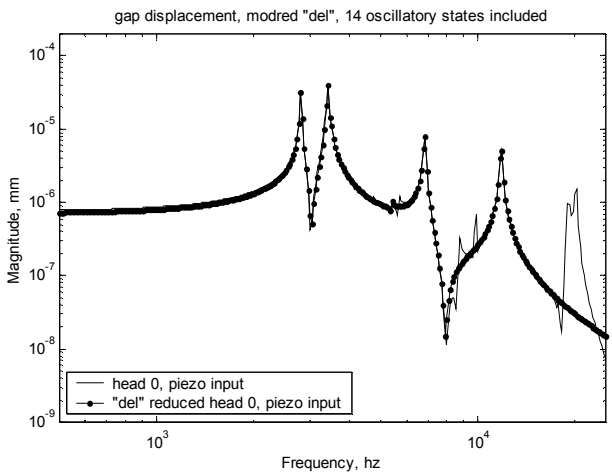

Figure 19.49: Head 0, piezo input, 14 reduced oscillatory states included.

For 14 oscillatory states included now the first four major piezo modes are fitted while the coil input starts missing some modes in the 10khz range.

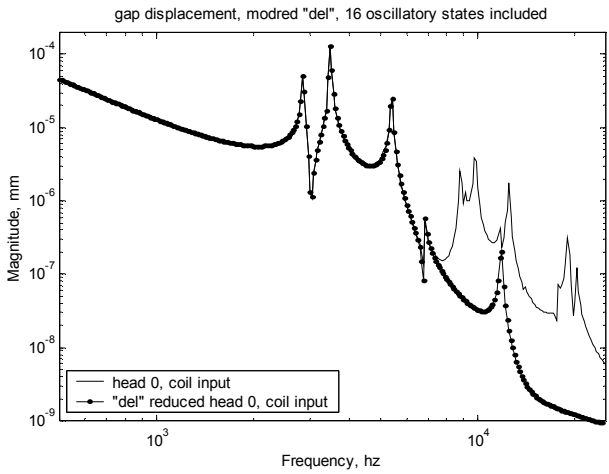

Figure 19.50: Head 0, coil input, 16 reduced oscillatory states included.

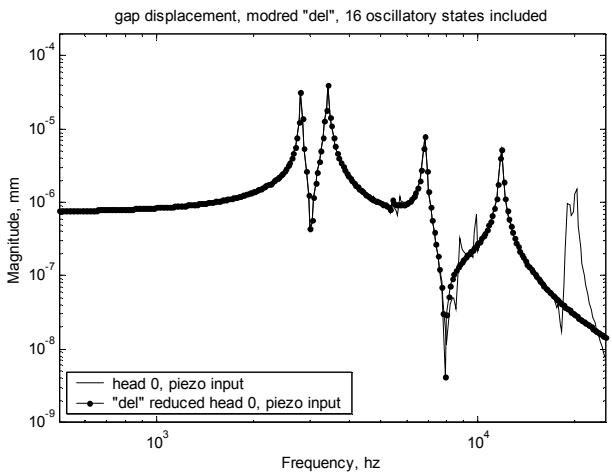

Figure 19.51: Head 0, piezo input, 16 reduced oscillatory states included.

For 16 oscillatory states included the only visible effect of the extra two states is in the piezo input zero in the 8khz range.

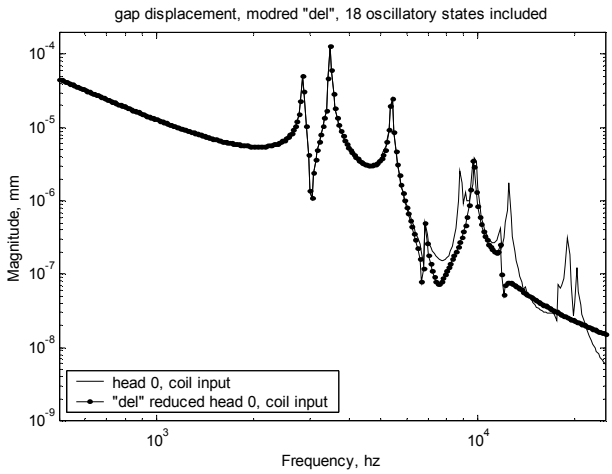

Figure 19.52: Head 0, coil input, 18 reduced oscillatory states included.

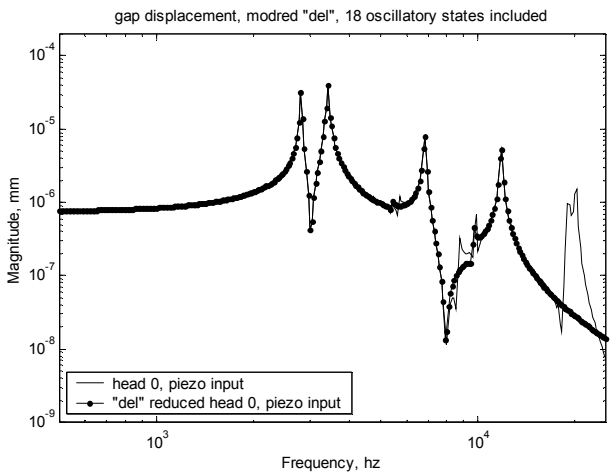

Figure 19.53: Head 0, piezo input, 18 reduced oscillatory states included.

For 18 oscillatory states included the coil input response picks up an additional mode in the 10khz range.

19.5.6 "del" and "mdc" Frequency Response Comparison

This section compares the "del" and "mdc" reduced models for the case of 20 included oscillatory states.

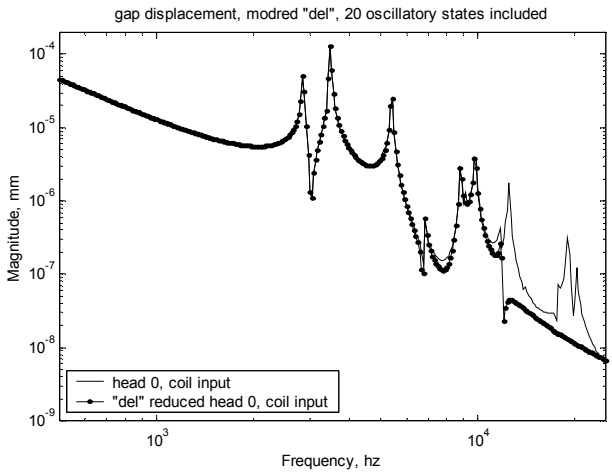

Figure 19.54: Head 0, coil input, 20 reduced oscillatory states included, modred "del."

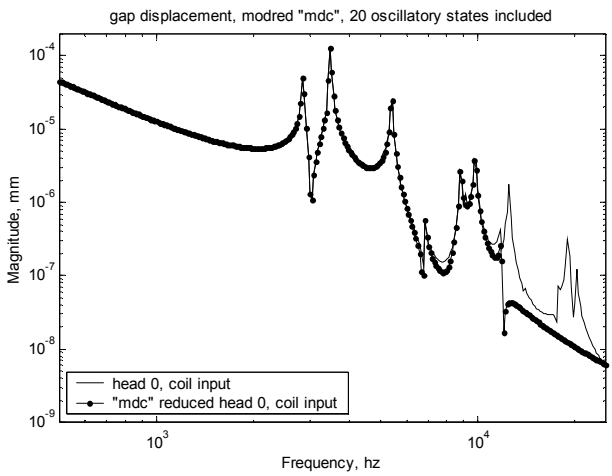

Figure 19.55: Head 0, coil input, 20 reduced oscillatory states included, modred "mdc."

There is virtually no difference between the "del" and "mdc" reductions in the two figures above for coil input.

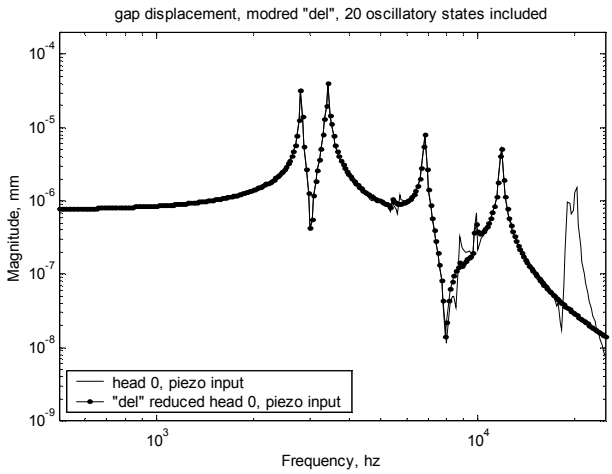

Figure 19.56: Head 0, piezo input, 20 reduced oscillatory states included, modred "del."

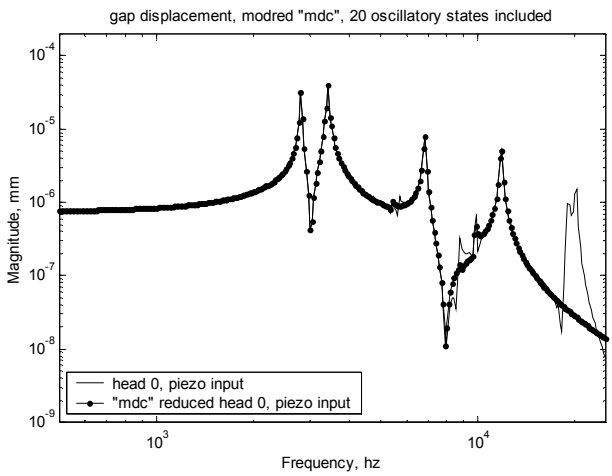

Figure 19.57: Head 0, piezo input, 20 reduced oscillatory states included, modred "mdc."

Similarly, there is no difference between the "del" and "mdc" reductions for piezo input.

19.5.7 Impulse Response

Oscillatory system impulse responses due to both coil and piezo forcing functions are calculated. Previously calculated results for normalized reduction index (18.28) versus number of modes included are shown.

```
%       calculate impulse responses

        ttotal = 0.0025;

        t = linspace(0,ttotal,400)';

        [disp_syso,t_syso] = impulse(syso,t);

        [disp_rsys_delo,t_rsys_delo] = impulse(rsys_delo,t);

        [disp_rsys_mdco,t_rsys_mdco] = impulse(rsys_mdco,t);

        disph0coil = disp_syso(:,2,1);
        disph1coil = disp_syso(:,1,1);
        disph0pz   = disp_syso(:,2,2);
        disph1pz   = disp_syso(:,1,2);

        dispr_delh0coil = disp_rsys_delo(:,2,1);
        dispr_delh1coil = disp_rsys_delo(:,1,1);
        dispr_delh0pz   = disp_rsys_delo(:,2,2);
        dispr_delh1pz   = disp_rsys_delo(:,1,2);

        dispr_mdch0coil = disp_rsys_mdco(:,2,1);
        dispr_mdch1coil = disp_rsys_mdco(:,1,1);
        dispr_mdch0pz   = disp_rsys_mdco(:,2,2);
        dispr_mdch1pz   = disp_rsys_mdco(:,1,2);

%       build matrix of results

        dispo = [disph0coil disph1coil disph0pz disph1pz ...
                 dispr_delh0coil dispr_delh1coil dispr_delh0pz dispr_delh1pz ...
                 dispr_mdch0coil dispr_mdch1coil dispr_mdch0pz dispr_mdch1pz];

        h0coil_del_del =  dispo(:,1) - dispo(:,5);

        h1coil_del_del =  dispo(:,2) - dispo(:,6);

        h0piezo_del_del = dispo(:,3) - dispo(:,7);

        h1piezo_del_del = dispo(:,4) - dispo(:,8);

        h0coil_mdc_del =  dispo(:,1) - dispo(:,9);

        h1coil_mdc_del =  dispo(:,2) - dispo(:,10);

        h0piezo_mdc_del = dispo(:,3) - dispo(:,11);

        h1piezo_mdc_del = dispo(:,4) - dispo(:,12);

        index_h0coil_del =  …
                sqrt(sum(h0coil_del_del.*h0coil_del_del))/sqrt(sum(dispo(:,1).*dispo(:,1)));

        index_h1coil_del = …
                sqrt(sum(h1coil_del_del.*h1coil_del_del))/sqrt(sum(dispo(:,2).*dispo(:,2)));
```

```
index_h0piezo_del = …
       sqrt(sum(h0piezo_del_del.*h0piezo_del_del))/sqrt(sum(dispo(:,3).*dispo(:,3)));

index_h1piezo_del = …
       sqrt(sum(h1piezo_del_del.*h1piezo_del_del))/sqrt(sum(dispo(:,4).*dispo(:,4)));

index_h0coil_mdc = …
      sqrt(sum(h0coil_mdc_del.*h0coil_mdc_del))/sqrt(sum(dispo(:,1).*dispo(:,1)));

index_h1coil_mdc = …
      sqrt(sum(h1coil_mdc_del.*h1coil_mdc_del))/sqrt(sum(dispo(:,2).*dispo(:,2)));

index_h0piezo_mdc = …
     sqrt(sum(h0piezo_mdc_del.*h0piezo_mdc_del))/sqrt(sum(dispo(:,3).*dispo(:,3)));

index_h1piezo_mdc = …
     sqrt(sum(h1piezo_mdc_del.*h1piezo_mdc_del))/sqrt(sum(dispo(:,4).*dispo(:,4)));

[index_h0coil_del index_h1coil_del index_h0piezo_del index_h1piezo_del ...
index_h0coil_mdc index_h1coil_mdc index_h0piezo_mdc index_h1piezo_mdc]

plot(t_syso,disph0coil,'k.-',t_rsys_delo,dispr_delh0coil, …
                'k-',t_rsys_mdco,dispr_mdch0coil,'k--')
title(['head 0, displacement vs time, coil impulse input, ', …
            num2str(osc_states_used),' oscillatory states included'])
xlabel('time, sec')
ylabel('displacement, mm')
legend('all modes','modred del','modred mdc',4)
grid off

disp('execution paused to display figure, "enter" to continue');%pause

plot(t_syso,disph1coil,'k.-',t_rsys_delo,dispr_delh1coil, …
            'k-',t_rsys_mdco,dispr_mdch1coil,'k--')
title(['head 1, displacement vs time, coil impulse input, ', …
           num2str(osc_states_used),' oscillatory states included'])
xlabel('time, sec')
ylabel('displacement, mm')
legend('all modes','modred del','modred mdc',4)
grid off

disp('execution paused to display figure, "enter" to continue');%pause

plot(t_syso,disph0pz,'k.-',t_rsys_delo,dispr_delh0pz, …
                'k-',t_rsys_mdco,dispr_mdch0pz,'k--')
title(['head 0, displacement vs time, piezo impulse input, ', …
                num2str(osc_states_used),' oscillatory states included'])
xlabel('time, sec')
ylabel('displacement, mm')
legend('all modes','modred del','modred mdc',4)
grid off

disp('execution paused to display figure, "enter" to continue');%pause
```

```
plot(t_syso,disph1pz,'k.-',t_rsys_delo,dispr_delh1pz, …
                'k-',t_rsys_mdco,dispr_mdch1pz,'k--')
title(['head 1, displacement vs time, piezo impulse input, ', …
                num2str(osc_states_used),' oscillatory states included'])
xlabel('time, sec')
ylabel('displacement, mm')
legend('all modes','modred del','modred mdc',4)
grid off

disp('execution paused to display figure, "enter" to continue');%pause

%                   states  h0cd    h1cd    h0pd    h1pd    h0cm    h1cm    h0pm    h1pm

error = [    10  0.1081  0.1075  0.4162  0.3963  0.1081  0.1075  0.4165  0.3964

                    12  0.1079  0.1072  0.3154  0.3058  0.1079  0.1073  0.3157  0.3061

                    16  0.1075  0.1070  0.1393  0.1421  0.1074  0.1070  0.1393  0.1419

                    20  0.0395  0.0425  0.1391  0.1410  0.0397  0.0425  0.1391  0.1411

                    24  0.0363 0.0374  0.0839  0.0873  0.0463  0.0473  0.0841  0.0875

                    28  0.0161  0.0178  0.0469  0.0495  0.0160  0.0191  0.0791  0.0794

                    32  0.0140  0.0142  0.0145  0.0160  0.0142 0.0143  0.0146  0.0163];

nmode = error(:,1)/2;

error_h0coil_del = error(:,2);

error_h1coil_del = error(:,3);

error_h0piezo_del = error(:,4);

error_h1piezo_del = error(:,5);

error_h0coil_mdc = error(:,6);

error_h1coil_mdc = error(:,7);

error_h0piezo_mdc = error(:,8);

error_h1piezo_mdc = error(:,9);

plot(nmode,error_h0coil_del,'k.-',nmode,error_h0coil_mdc,'k-')
title('head 0, coil input normalized reduction index')
xlabel('number of modes included')
ylabel('normalized reduction index')
legend('modred del','modred mdc')
axis([0 20 0 0.5])
grid off

disp('execution paused to display figure, "enter" to continue');%pause
```

```
plot(nmode,error_h1coil_del,'k.-',nmode,error_h1coil_mdc,'k-')
title('head 1, coil input normalized reduction index')
xlabel('number of modes included')
ylabel('normalized reduction index')
legend('modred del','modred mdc')
axis([0 20 0 0.5])
grid off

disp('execution paused to display figure, "enter" to continue');%pause

plot(nmode,error_h0piezo_del,'k.-',nmode,error_h0piezo_mdc,'k-')
title('head 0, piezo input normalized reduction index')
xlabel('number of modes included')
ylabel('normalized reduction index')
legend('modred del','modred mdc')
axis([0 20 0 0.5])
grid off

disp('execution paused to display figure, "enter" to continue');%pause

plot(nmode,error_h1piezo_del,'k.-',nmode,error_h1piezo_mdc,'k-')
title('head 1, piezo input normalized reduction index')
xlabel('number of modes included')
ylabel('normalized reduction index')
legend('modred del','modred mdc')
axis([0 20 0 0.5])
grid off

disp('execution paused to display figure, "enter" to continue');%pause
```

The pages following will show impulse responses for head 0 for both coil and piezo inputs and for both “del” and “mdc” reduced models. Following the impulse responses, the normalized reduction index versus number of reduced modes is plotted. It shows very little difference between the two reduction methods.

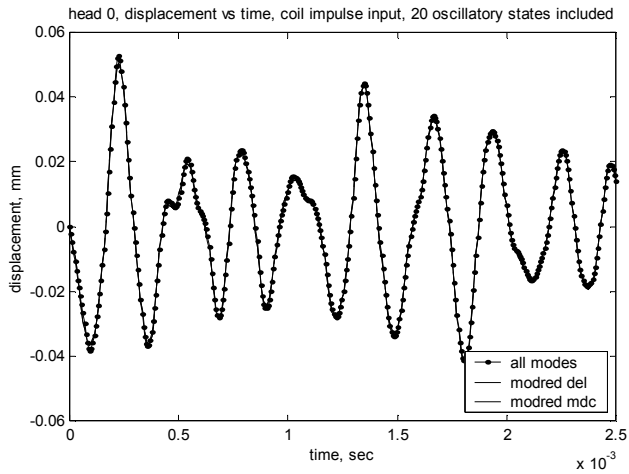

Figure 19.58: Impulse response comparison for head 0 for coil input for oscillatory system, full model (all oscillatory modes) and balreal modred "del" and "mdc" reduced systems with 20 oscillatory modes.

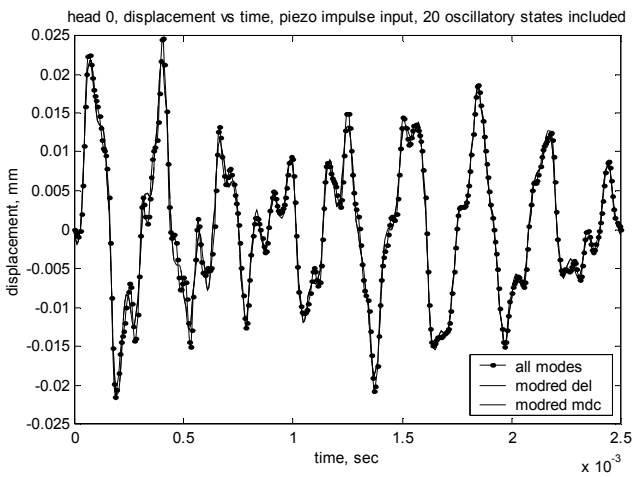

Figure 19.59: Impulse response comparison for head 0 for piezo input for oscillatory system, full model (all oscillatory modes) and balreal modred "del" and "mdc" reduced systems with 20 oscillatory modes.

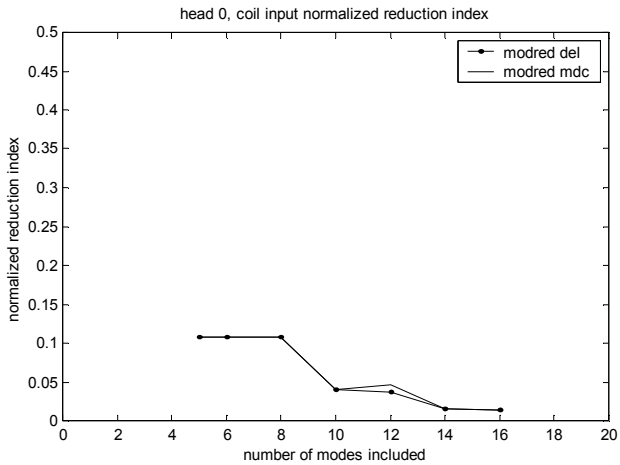

Figure 19.60: Head 0 impulse response normalized error index comparison for reduced modred models using "del" and "mdc" methods, coil input.

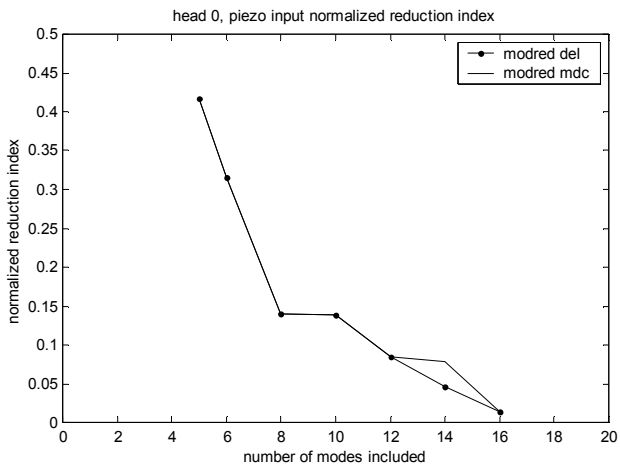

Figure 19.61: Head 0 impulse response normalized error index comparison for reduced modred models using "del" and "mdc" methods, piezo input.

19.6 MIMO Summary

We started the chapter with a description of key mode shapes for the two-stage actuator/suspension system. ANSYS eigenvector listings for several modes allowed comparing the numeric values in the eigenvector to the visual interpretation from the mode shape plot. Small displacements in the deformed mode shape plot correlate to small numerical values in the eigenvector. If the small numerical values in the eigenvector occur in the input and/or output degrees of freedom, the mode will have a "small" dc gain and is relatively unimportant.

In the next section we calculated and plotted the dc gains for all four input/output combinations. In Table 19.1 we listed the modes for the input/output combinations, sorted by dc gain. We found that head 0 and head 1 dc gain sorted modes for coil input are the same for the first seven modes. For piezo input, both heads have the same mode ranking for the first six modes. This similarity in the most important modes for both heads for the coil and piezo inputs is brought about by the physical symmetry of the actuator/suspension system, and in general will not be the case.

As in the previous chapter, we used balancing to define the system for reduction and used the "modred" "del" and "mdc" options to reduce. Frequency responses for different number of states were plotted and compared for both coil and piezo inputs, overlaying the non-reduced transfer function.

Visually comparing the reduced and non-reduced frequency response magnitudes, we found that including 20 oscillatory states (plus the states from the one rigid body mode) gave a "good" fit through the 10khz range.

The MATLAB model was then used to calculate the impulse responses for the oscillatory reduced and non-reduced systems, where we found that 10 oscillatory modes (20 oscillatory states) were required to have a normalized error index of less than 5% for coil inputs. For piezo inputs, 16 oscillatory modes (32 oscillatory states) were required for less than 5% normalized error index. There was little difference in normalized error index between the "del" and "mdc" reduction options.

Problems

P19.1 Modify the MATLAB code **act8pz.m** to reduce the piezo force "fpz" (Section 19.5.2) from the 0.2 value used in the text to 0.02 and 0.002. In both cases, examine the frequency and impulse responses for different number of oscillatory states used. Does the balanced reduction method technique continue to choose roughly equal number of modes for both coil and piezo inputs even when there are large differences in dc gain values between the two inputs?

P19.2 For the piezo force "fpz" of 0.2, choose the first five oscillatory modes from the coil input and the first five oscillatory modes from the piezo input (Table 19.1). Assemble the state equations from the rigid body mode and the 10 oscillatory modes and solve for the frequency and impulse responses. Compare the responses to the 20 oscillatory state balanced reduction. Comment on the similarities/differences.