ROTORDYNAMIC ANALYSIS OF THEORETICAL MODELS AND EXPERIMENTAL SYSTEMS

A Thesis

presented to

the Faculty of California Polytechnic State University,

San Luis Obispo

In Partial Fulfillment

of the Requirements for the Degree

Master of Science in Mechanical Engineering

by

Cameron Naugle

April 2018

© 2018
Cameron Naugle
ALL RIGHTS RESERVED

COMMITTEE MEMBERSHIP

TITLE:	Rotordynamic Analysis of Theoretical Models and Experimental Systems
AUTHOR:	Cameron Naugle
DATE SUBMITTED:	April 2018
COMMITTEE CHAIR:	Xi (Julia) Wu, Ph.D. Professor of Mechanical Engineering
COMMITTEE MEMBER:	Mohammad Noori, Ph.D. Professor of Mechanical Engineering
COMMITTEE MEMBER:	Peter Schuster, Ph.D. Professor of Mechanical Engineering

ABSTRACT

Rotordynamic Analysis of Theoretical Models and Experimental Systems

Cameron Naugle

This thesis is intended to provide fundamental information for the construction and analysis of rotordynamic theoretical models, and their comparison the experimental systems. Finite Element Method (FEM) is used to construct models using Timoshenko beam elements with viscous and hysteretic internal damping. Eigenvalues and eigenvectors of state space equations are used to perform stability analysis, produce critical speed maps, and visualize mode shapes. Frequency domain analysis of theoretical models is used to provide Bode diagrams and in experimental data full spectrum cascade plots. Experimental and theoretical model analyses are used to optimize the control algorithm for an Active Magnetic Bearing on an overhung rotor.

ACKNOWLEDGMENTS

Thanks to:

- Bently Nevada for their support through the Donald E. Bently Center for Engineering Innovation at Cal Poly. The equipment provided by Bently Nevada made this work possible.
- my Advisor, Xi Wu, for always thinking big, and for supporting all my work.
- my family and friends for their support and love.

TABLE OF CONTENTS

Page

LIST OF TABLES viii

LIST OF FIGURES ix

CHAPTER

1 Vibration Signal Analysis 1

1.1 Data Collection and Processing 1

1.1.1 Amplitude 4

1.1.2 Spectrum 4

1.1.3 Phase 6

1.1.3.1 Time Domain Approach 6

1.1.3.2 Frequency Domain Approach 7

1.2 Rotordynamic Figures 8

1.2.1 Bode 8

1.2.2 Full Spectrum and Full Spectrum Cascade 10

1.2.3 Orbit 11

1.2.4 Filtering 13

1.2.5 Frequency and Time Resolution 14

2 Finite Element Method For Rotordynamic systems 18

2.1 Timoshenko Beam Finite Element 18

2.1.1 Kinematic Relationships 19

2.1.2 Internal Constitutive Relationship 22

2.1.3 Differential Equations of Motion 25

2.1.4 Shape Functions 29

2.1.5 Finite Equations of Motion 34

2.1.6 Rotating Internal Damping 34

2.1.7 Beam Element in Complex Coordinates 37

2.2 Disk Nodal Equations 39

2.2.1 Disk in complex coordinates 40

2.3 Bearing Nodal Equations 40

2.3.1 Bearing in Complex Coordinates . . . 41
2.4 Assembly of the Global System of Equations . . . 41
2.4.1 Assembly In the Real Coordinate System . . . 42
2.4.2 In the Complex Coordinate System . . . 42
2.5 Analysis of the Resulting Model . . . 42
3 Frequency Domain Analysis . . . 44
3.1 State Space Representation and the Eigenvalue Problem . . . 45
3.2 Dynamic Response . . . 47
3.3 Roots Locus and Stability Analysis . . . 50
3.4 Campbell . . . 53
3.5 Shapes . . . 54
4 Synthesis in Example of a Magnetic Bearing on an Overhung Rotor . . . 58
4.1 Physical System Description . . . 58
4.2 Experimental Results . . . 59
4.3 Theoretical Model . . . 61
4.3.1 Active Magnetic Bearing . . . 65
4.3.1.1 Proportional Derivative Control . . . 67
4.4 Addition of Magnetic Bearing to the Rotor Model . . . 68
5 Conclusion . . . 74
5.1 Summary . . . 74
5.2 Future Work . . . 74
BIBLIOGRAPHY . . . 76
APPENDICES
A Bernoulli-Euler Beam equation . . . 79
B RotorFEM MATLAB Code for Constructing and Analyzing FEM Models 82
B.1 Main Object File . . . 82
B.2 Matrix Assembly File . . . 82
B.3 Elemental Matrices . . . 84
B.4 Plotting Functions . . . 87
C RotorDAQ MATLAB Code for Analyzing Vibration Signals . . . 97
C.1 Main Object File . . . 97
C.2 Plotting Functions . . . 101

LIST OF TABLES

Table		Page
3.1	Properties of disks, shaft elements, and bearings of the example problem.	46
4.1	Geometric parameters of the overhung rotor system.	59
4.2	Properties of disks, shaft elements, and bearings of the theoretical model.	62
4.3	Active Magnetic Bearing Parameters.	71

LIST OF FIGURES

Figure		Page
1.1	Position of the rotor shaft over a period in time.	2
1.2	Rotor Spin Speed Windowing effect.	3
1.3	A window in time of the transient vibration signals for a window, N, with nspw of 2048[samples].	4
1.4	Bode diagram of the experimental example overhung system. Signals are filtered to synchronous speed.	9
1.5	Example Full Spectrum of experimental example system at Ω = 1500[RPM].	10
1.6	Cascade of the experimental system described in §1.2.	11
1.7	3D Orbit of the experimental system described in §1.2. Lighter colors indicate larger vibration.	12
1.8	Orbits of the experimental example. Spin speed is counterclockwise. Dots indicate the reference position of the shaft, and the beginning of each orbit.	13
1.9	Cascade of the experimental system with a synchronous filter applied.	14
1.10	Cascade of the experimental system with an f_{res} of 0.2[Hz] and a resulting Ω_{res} of 54[RPM].	16
1.11	Cascade of the experimental system with an f_{res} of 2[Hz] and a resulting Ω_{res} of 5.5[RPM].	17
2.1	Timoshenko beam section with degrees of freedom at some point x along beam axis.	19
2.2	Beam Element with nodal displacements.	20
2.3	Beam differential element with generalized forces.	26
2.4	Beam Element with nodal displacements.	30
2.5	Shape Functions as they vary with ζ using two different ratios of length to radius of beam element.	33
3.1	Diagram of Two disk model example problem.	45
3.2	Bode diagram of the second disk subject to an unbalance at the second disk. Amplitudes of v & w are identical for this asymmetric system, phase of w would be lagging v by 90[deg].	48

3.3 Nyquist plots for the first two modes at node 6 for the example two disk problem. 49
3.4 Nyquist Diagram for the first mode at node 6 in the whirl speed range $1100 < \omega < 1300$[RPM] at a spin speed, $\Omega = 4000$[RPM]. Counter-clockwise path indicates instability. 49
3.5 Roots Locus of the example problem with an internal damping coefficient of $0.0002[s]$. 51
3.6 Roots Locus of the example problem in 3-D with an internal damping coefficient of $0.0002[s]$. 52
3.7 Stability plot of the example problem. 53
3.8 Damping ratio of the first three modes of the example problem, with indication of threshold of stability for each mode. 54
3.9 Campbell Diagram of the example problem. 55
3.10 Modal shapes of the example problem. 56

4.1 Overhung rotor system diagram. 58
4.2 3D Orbit of the experimental overhung rotor system. 59
4.3 Cascade of the experimental overhung rotor system. 60
4.4 Bode diagram of the experimental overhung rotor filtered to 1X. . . 61
4.6 Bode Diagram of overhung system without AMB. 64
4.7 Campbell Diagram of the overhung system without AMB. 64
4.8 Roots Locus of overhung system without AMB. 65
4.11 Roots locus of Overhung rotor system with varying k_v. 72
4.12 Bode diagram at node 6 comparing the rotor without AMB(solid) and with AMB(dashed). 73
4.13 Bode diagram at node 6 comparing the rotor without AMB(solid) and with AMB(dashed) for complete levitation at node 4. 73

A.1 Free body diagram of a beam section in planar bending. 80

Chapter 1

VIBRATION SIGNAL ANALYSIS

1.1 Data Collection and Processing

The most important measurement to be made in order to perform significant rotordynamic analysis is the Vibration Signal, V or W. Other important measurements include Spin Speed, Ω, of the rotating shaft (especially important if during a start-up or run-down), and a Reference Signal, R, that indicates a rotational position of the shaft. Orthogonal vibration signals (meaning two independent directions), V & W, measuring the position of the shaft centerline can help in characterizing anisotropic systems. Sampling Rates, f_s of the signals mentioned thus far must be high enough to measure the vibration of interest, typically this is at least several times the highest expected spin speed of the shaft. It is important to note that these signals can come from an experiment or a theoretical model. In the case of a theoretical model, the sampling rate is inversely proportional to the time interval of the differential equation solver. Or, if a closed form solution exists, the time interval of the time vector chosen to express the solution within.

There are four variables deduced from the above signals that form the basis for the majority of rotordynamic figures and analysis. These are: Amplitude of Vibration, $A[m, mils]$; Amplitude Spectrum, $\tilde{A}(\omega)[m, mils]$; Phase of Vibration, $\beta[\text{deg}, Rad]$; and Spin Speed, $\Omega[Hz, Rad/s, RPM]$. Amplitude Spectrum is actually a two dimensional variable where the variable, $\omega[Hz, Rad/s, RPM]$, represents the Frequency of Vibration (often called Whirl Speed if its in units of rotation). Figures to be presented in this work are varying combinations of these four variables. The Bode diagram plots A against Ω alongside β against Ω. A Spectrum figure plots the absolute value of

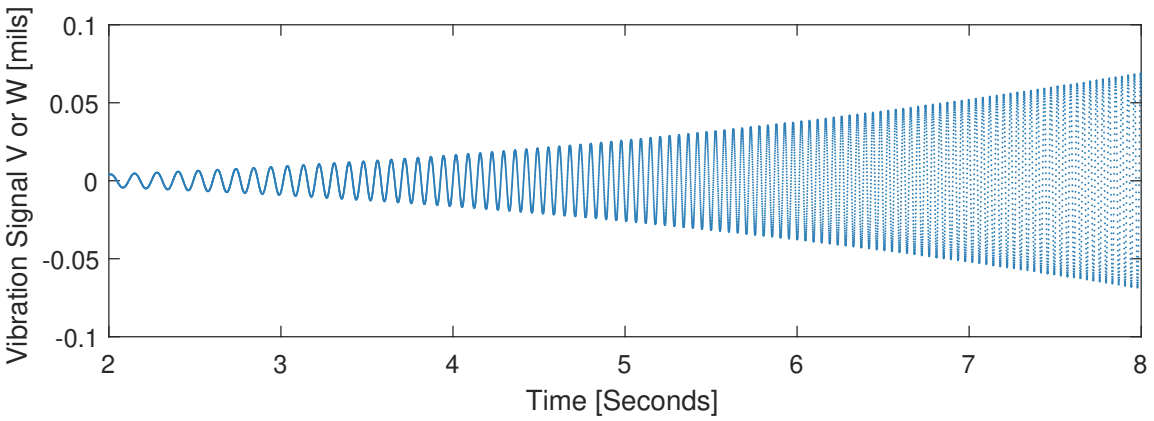
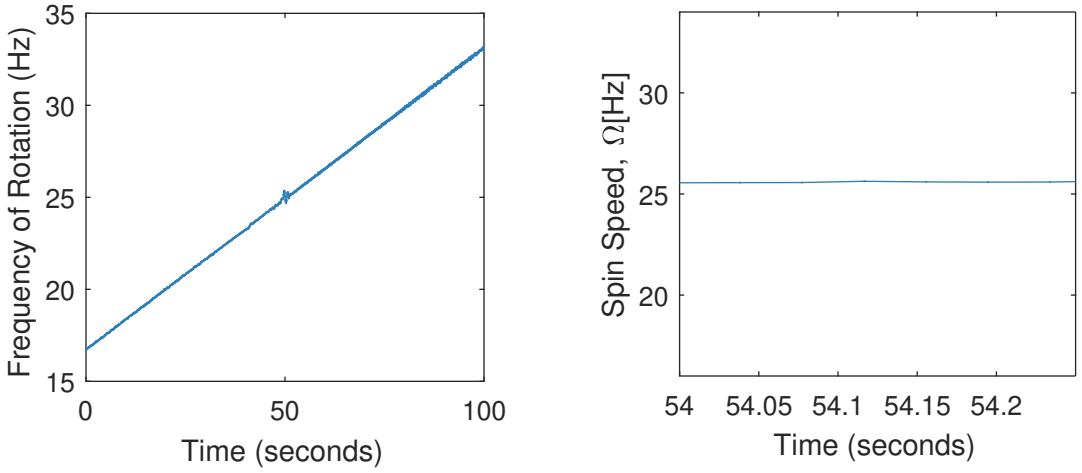


Figure 1.1: Position of the rotor shaft over a period in time.

$\tilde{A}(\omega)$. An extension of this is the Cascade which adds the third dimension of Ω. Orbit plots represent one cycle of the time domain signals V & W. Lastly, 3D Orbits are formed by plotting the orbit (V, W) against Ω. Explanations of the importance and methods in producing these plots are to follow.

A difficulty to rotordynamic analysis arises in the continuous change of Ω causing a continuous change in time of its dependent variables $A,\ \tilde{A}(\omega),\ \&\ \beta$. A visualization of these changes is given in Figure 1.1. This transient nature poses difficulty because the techniques used to produce $A,\ \tilde{A}(\omega),\ \&\ \beta$ from $V,\ W \&\ R$ rely on a span of subsequent rotations. A solution to this dilemma utilized in this work is the discretization of signals $V,\ W,\ R\ \&\ \Omega$ into windows in time. Window width will be represented by the variable $nspw[samples]$ and total number of samples divided by $nspw$ will give the number of windows in the signals, NW. There is a trade off between resolution in time, and resolution in frequency of variables as $nspw$ is changed. This trade off will be elaborated on in §1.2.5.

The windowing approach is visually depicted using Ω in figure 1.2 as it changes over time. When the width of the window is small enough, such as in Figure 1.2b, the change in the dependent variable becomes vanishingly small. Other continuous



(a) Rotor speed change over time during a ramp up.

(b) Rotor speed change over time during a ramp up.

Figure 1.2: Rotor Spin Speed Windowing effect.

variables such as A, $\tilde{A}(\omega)$, & β are expected to behave similarily, in that their value may be approximated by a single number, or in the case of $\tilde{A}(\omega)$ as a single spectrum, inside the window despite the change in that variable throughout the entire length of time. The variables $V(t)$, $W(t)$& $R(t)$ will now take the form $V(n)$, $W(n)$& $R(n)$ inside the window (fig. 1.3), where n is sample number. Spin Speed is simply taken as an average in the window, $\Omega(N) = avg(\Omega(0 : nspw))$, where N is the current window index. Therefore, no further explanation is given for its determination. Amplitude, Phase, Amplitude Spectrum, and Spin Speed must be calculated in each window, for a series of windows that cover the entire length of signals. Then a vector of each variable will exist where the length is equal to the number of windows, NW, and is given by

$$NW = \frac{length(signals)}{nspw}$$

The calculation of each of the variables inside the window, at some N, is given in the following sections. A useful visual representation of the windowed signals

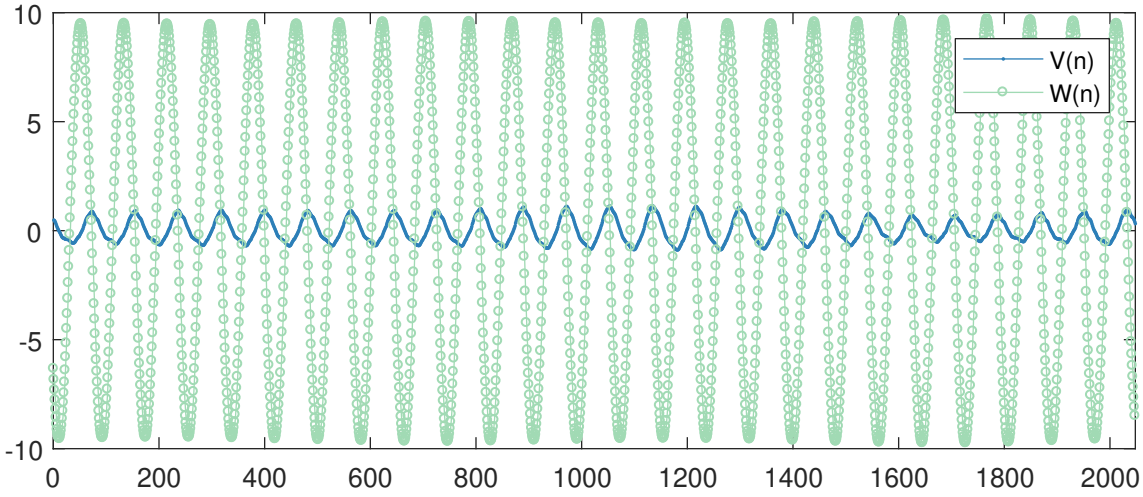


Figure 1.3: A window in time of the transient vibration signals for a window, N, with nspw of 2048[samples].

$V(n)$ &$W(n)$ are presented in Figure 1.3 to aid in understanding the sections to follow.

1.1.1 Amplitude

Amplitude calculation is fairly straightforward within the window. One approach to calculate the peak to peak amplitude is to take the average over the whole window, $A_v(N) = max(V(0 : nspw)) - min(V(0 : nspw))$. Another is to use a peak-finding algorithm to determine the height of each peak and average them all over the sample length. General computer code packages, such as MATLAB, will contain a peak finding algorithm, the details of which are out of the scope of this work.

1.1.2 Spectrum

The frequency spectrum of the signal is calculated inside the window using a Fourier Transform. Influence for this representation of complex amplitude spectrum comes from [4]. In MATLAB the Fast Fourier Transform (fft) has been preprogrammed

allowing easy working between time and frequency domains.

$$\tilde{A}_V(n) = \frac{\text{fft}(V(n))}{nspw}, \ \& \ \tilde{A}_W(n) = \frac{\text{fft}(W(n))}{nspw} \tag{1.1}$$

A useful way to represent the data is using a complex variable to compact the two orthogonal displacements V & W as

$$Z(n) = V(n) + iW(n) \tag{1.2}$$

now the spectrum of this complex value represents both equation planes of vibration in one equation:

$$\tilde{A}_{\pm}(n) = \frac{\text{fft}(Z(n))}{nspw} \tag{1.3}$$

Thus far, the frequency spectrum in the real coordinates and in the complex is in terms of samples on the independent axis. The frequency vector to which the fft() corresponds must be calculated–this is the whirl speed, ω. It is known that the slope of ω is $d\omega = f_s/nspw$. This value is also called the frequency resolution, f_{res}. It is also known that the frequency vector is the same length as the time domain signal

$$f = d\omega(0 : nspw - 1)$$

and to center the spectrum at a frequency of 0

$$Q = ceiling((nspw + 1)/2)$$
$$f_Q = d\omega(Q - 1)$$
$$\omega_j = f - f_Q$$

where ω_j, here in Hz, is the variable that pairs with the real or complex Amplitude spectrum and the subscript j is a reminder that ω is a discrete variable that ranges $1 < j < nspw$. Now the complex amplitude spectrums can be represented as $\tilde{A}_V(\omega), \ \tilde{A}_W(\omega), \ \& \ \tilde{A}_{\pm}(\omega)$. The Amplitude spectrum in real coordinates(1.1) is symmetric for positive and negative ω, so typically when the spectrum of a single signal

is presented in a spectrum plot or in a cascade it is only on the positive frequency side. On the other hand, the complex representation of the amplitude spectrum(1.3) is not symmetric on the positive and negative sides of ω. Understanding why this is the case stems from realizing the form of the Fourier transform as the summation of circles in the complex plane, $Z(n) = \sum_{j=1}^{j=nspw} \tilde{A_{\pm}}(\omega_j)e^{i\omega_j t}$. When ω_j is positive this represents a positive rotation, and when negative represents a negative rotation. For a given whirl speed, ω_j, the positive of that value will be represented by $+\omega$ and the negative as $-\omega$. These two representations of ω correspond to two separate indexes j as ω_j is symmetric about $j = nspw/2$. This results in the sum of a positively rotating circle of amplitude, $\tilde{A}(+\omega)$ and a negatively rotating circle of amplitude $\tilde{A}(-\omega)$. The ellipse formed by this summation is the orbit of the shaft centerline at this specific speed[13],[2]. With the understanding of contributions of $\tilde{A}(\omega)$ and $\tilde{A}(-\omega)$, we realize that the resulting ellipse will rotate in the counterclockwise direction if $\tilde{A}(\omega) > \tilde{A}(-\omega)$ and in the clockwise direction otherwise. This rotation is in reference to the positive rotation about the z axis defined by the right hand rule from the y-z plane. Since the same coordinate system is used to represent the sign of Ω, the whirl can be interpreted as in or opposed to the direction of spin. For a positive Ω, a negative ω corresponds with an opposing whirl, and vice-versa for a positive ω.

1.1.3 Phase

1.1.3.1 Time Domain Approach

A rather direct way of calculating the phase angle comes from an inspection of the time domain signal. If some once-per-turn reference is available, then a zero-crossing, peak-finding, or threshold algorithm can be employed to locate a specific reference angle of the shaft rotation. In the case that the vibration signal is mostly synchronous (vibrating at the same frequency as the rotation of the shaft) then a peak-finding or

zero-crossing algorithm can be used to determine the number of samples from the shaft reference angle to the peak of the vibration. Comparison of this sample distance to the sample distance of an entire cycle of the reference signal will reveal the amount a signal lags the reference angle as a portion of a full rotation. In terms of samples this can be represented by the equation

$$\beta_k = 2\pi \frac{\#ref_k - \#peak_k}{\#ref_k - \#ref_{k-1}} \tag{1.4}$$

where β_k is the phase lag of the signal of interest from the reference signal at the kth reference cycle, $\#ref_k$ is the sample number of the reference trigger, and $\#peak_i$ is the sample number of the peak of the signal of interest. One large advantage to this brute force method is that it can run continuously and provide phase information on just the last rotation of the shaft. In the application to the window of vibration data, fig. 1.3, the measurements of each cycle would be averaged across the window as $\beta(\omega) = avg(\beta_k)$. This would be done for however many indexes k were found in the window. In the case of windowed data in Figure 1.3 $k = 7[rotations]$.

1.1.3.2 Frequency Domain Approach

Alternatively, the phase angle can be determined using the frequency domain representation of the signals. If the speed of the rotor is known and the time domain signals V & W are known to be synchronous, or filtered to synchronous, then the spectrums of the signals of interest can be used to calculate the phase delay. For any frequency, ω, the angle is calculated using the equation

$$\beta(\omega) = angle(\tilde{A}(\omega)) - angle(\tilde{K}(\omega))$$

where, $\tilde{K} = \frac{\text{fft}(K)}{nspw}$ is the frequency domain representation of the reference signal. Either $\tilde{A}_V$, or $\tilde{A}_W$ is used to find the phase delay of the V or W time domain vibration in reference to the once per turn reference of K. It is also possible to find the delay

of any time domain signals in reference to any other time domain signal at a specific frequency using the above equation, though the common practice is to compute the β angle of both signals and subtract one from another. In synchronous vibration, $\omega = \Omega$.

1.2 Rotordynamic Figures

In the previous section Amplitude, Phase, and Amplitude Spectrum were calculated for the interior of a window of index N. Each of these variables then need to be indexed, as all of the windows are processed until the entire signal has been exhausted. The total number of windows can be realized in the equation $NW = \frac{length(X)}{nspw}$ where X is a placeholder for any time domain signal. After all windows have been exhausted, vectors for $A(N)$, $\beta(N)$ & $\Omega(N)$ will all be of length NW, and $\tilde{A}(N, \omega)$ is a matrix of size $(NW, nspw)$.

For visualization of the plots, experimental data from a overhung rotor system, with one disk supported by two bushings, will used to demonstrate the figures in use. Data was taken during this experiment with two orthogonal position sensors, and a reference sensor providing shaft angle and speed information. The experiment consisted of a ramp-up from 1000[RPM] to 2000[RPM].

1.2.1 Bode

The Bode diagram for the example overhung rotor system is given in Figure 1.4. By looking at the amplitude portion of the plot it is evident that the V signal undergoes a natural frequency before the W signal, because the peak for A_V occurs before A_W. This idea is also supported through the inspection of the phase lag portion of the plot, as two seperate transitions are evident. Having two seperate peaks is an indication of high stiffness anisotropy in the system. By observing the phase lag of each signal, it

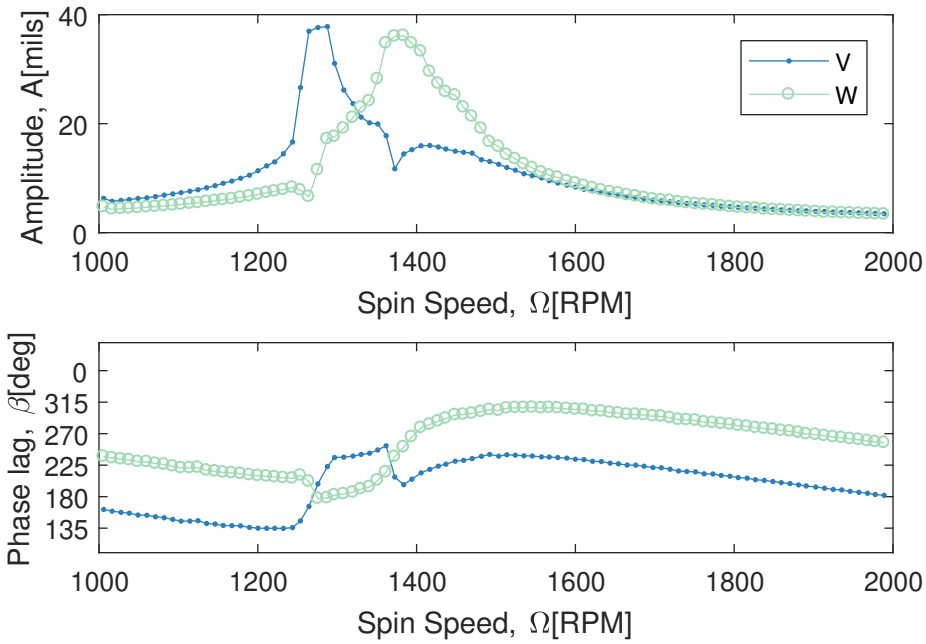


Figure 1.4: Bode diagram of the experimental example overhung system. Signals are filtered to synchronous speed.

is evident that the orbit direction is opposite the spin speed between speeds 1280-1350[RPM]. If normally W lags V with a positive counterclockwise rotation of the shaft (as is suggested by the phase angles in sub-synchronous and super-synchronous range), then during the critical speed, the orbit is reversed since V begins to lag W.

Bode diagrams are extremely useful in diagnosing system unbalance through inspection of phase lag information. If the shaft was perfectly straight before any deformation due to rotating unbalance, then the phase lag just before the first natural frequency is the angle of the unbalance vector. This is due to the fact that before the first natural frequency, the unbalance vector is aligned with the vibration radially out from the center of rotation. Furthermore, natural frequencies can be detected through the use of the phase lag information. Phase lag typically shifts 180[deg] after completely passing through a natural frequency, and is at 90[deg] during a natural frequency.

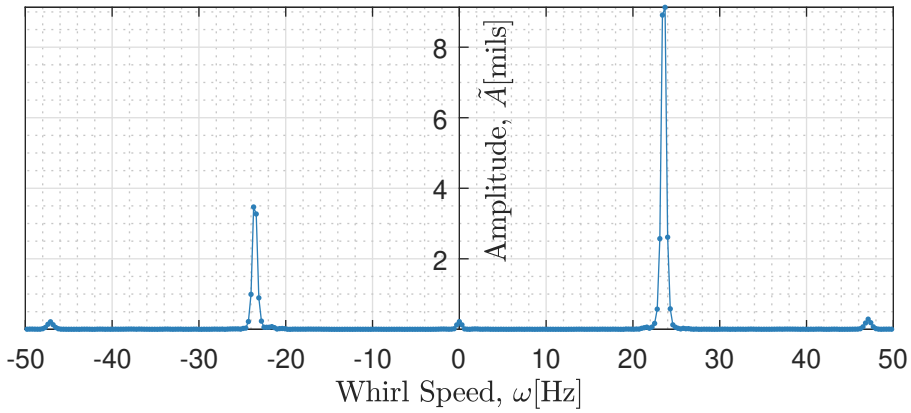


Figure 1.5: Example Full Spectrum of experimental example system at $\Omega = 1500$[RPM].

1.2.2 Full Spectrum and Full Spectrum Cascade

A single Complex Amplitude Spectrum at a specific speed, $\tilde{A}_{\pm}(1500[RPM], \omega)$ is shown as Figure 1.5. This complex representation of the spectrum is referred to as the "Full Spectrum" of the time domain signal because it contains both positive and negative frequencies. This figure tells us that the orbit at this speed is in the positive whirl direction at the dominant frequency, since the positive amplitude, $\tilde{A}_{\pm}(+\omega) = 9.13[mils]$ at its peak is greater than $\tilde{A}_{\pm}(-\omega) = 3.47[mils]$. Also, there is minimal amplitude in the spectrum other than this single frequency of $\pm 23.7[Hz]$ indicating the vibration is highly synchronous.

A Cascade plot is demonstrated with the experimental system described in §1.2, as Figure 1.6. Using this figure it is easy to detect the portion of the start-up in which the orbit is whirling opposite the spin speed. A sharp dip in positive amplitude, $\tilde{A}_{\pm}(\omega)$, correlated with a sharp rise in negative amplitude, $\tilde{A}_{\pm}(-\omega)$, at around 1320[RPM] leads to this phenomena.

The Cascade plot is particularly useful in characterizing non-synchronous vibration. Slightly evident in the example cascade and spectrum of figures 1.6, & 1.5 respectively, is the super-synchronous vibration at twice the spin speed, this is often

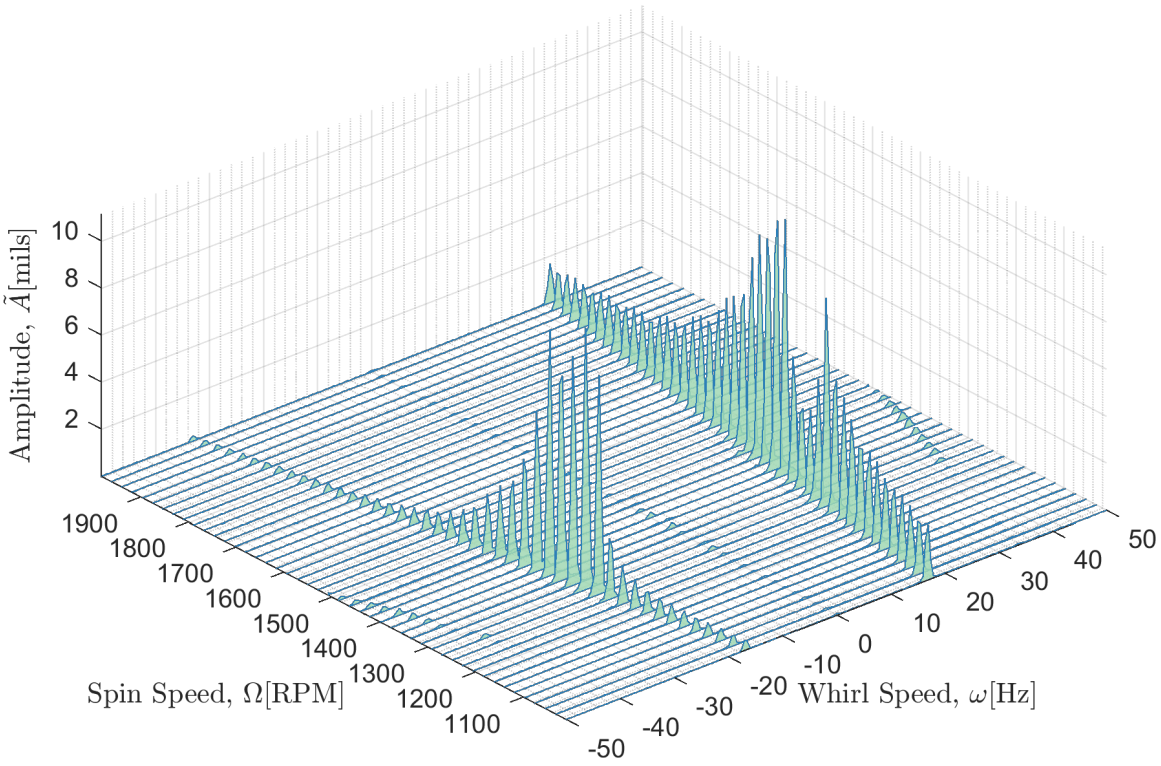


Figure 1.6: Cascade of the experimental system described in §1.2.

called the 2X vibration. Similarly to the 2X vibration, other non-synchronous whirl speeds can be referenced as nX where n is the multiple of synchronous. The cascade plot is an indispensable tool for the analysis of fluid film bearing, for example, as they are characterized by sub-synchronous whirl that is difficult to identify in other diagnostic diagrams. The cascade provides an overview of all system whirling at all spin speeds allowing identification of various multiples of synchronous speed. These super- or sub-synchronous whirl speeds of the form nX can then be used to filter the time domain signals, allowing isolation of specific dynamic phenomena.

1.2.3 Orbit

In the time domain, the actual orbit or trace of the centerline of the shaft is observed. In this work, the orbit is visualized in two ways: as a path in 2D space at a specific spin speed, or as a 3D orbit with a cascade of orbits as spin speed is increased. The

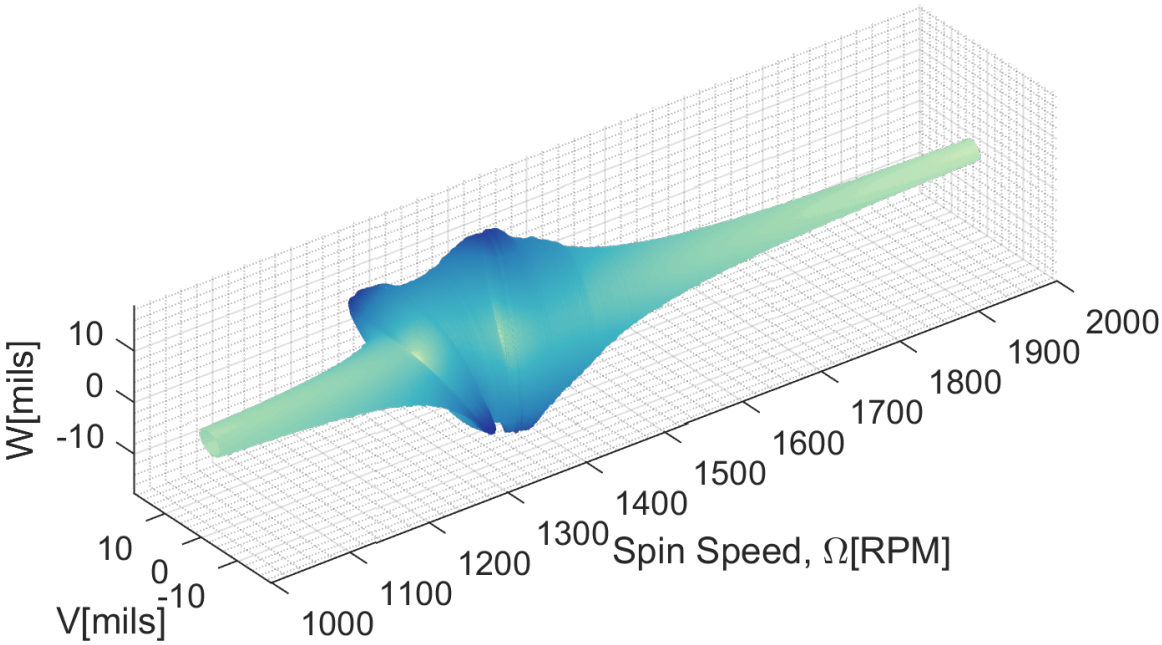


Figure 1.7: 3D Orbit of the experimental system described in §1.2. Lighter colors indicate larger vibration.

3D Orbit allows for the visualization of complicated phenomena in a simple intuitive way. Figure 1.7 3D Orbit is given for the experimental example described in §1.2. Appearing, once again is evidence of the negative whirl in the critical speed range. In the 3D orbit a collapsing of the shape can be seen between the speeds of 1200-1400[RPM] indicating the orbit has reversed its direction. Looking at independent orbits of specific speeds should explicitly demonstrate the orbit collapsing to a line, and turning negative. In Figure 1.8 at speed 1194$[RPM]$ the orbit is clearly whirling in the positive direction. As speed increases the orbit collapses into a line between speeds 1203 & 1206$[RPM]$ and begins whirling in the negative direction until the process is reversed by speed 1289$[RPM]$. Therefore, it can be confirmed that the orbit is whirling backward between the speeds 1205- 1280[RPM].

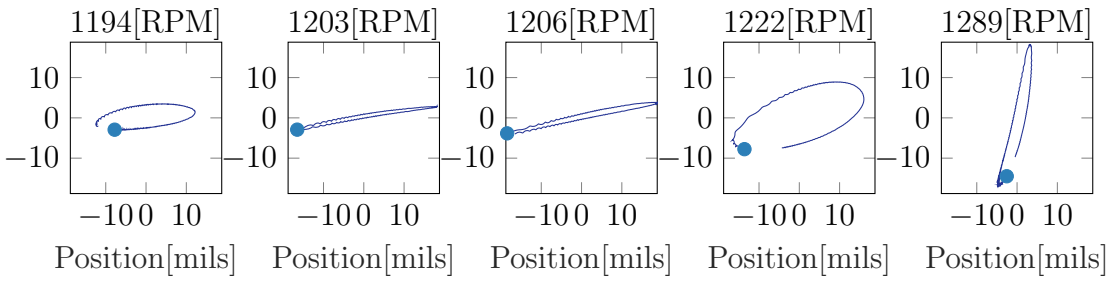


Figure 1.8: Orbits of the experimental example. Spin speed is counter-clockwise. Dots indicate the reference position of the shaft, and the beginning of each orbit.

1.2.4 Filtering

Correlating phase angles between signals can be extremely difficult due to the noise and harmonic frequencies that may disrupt the measurement of a phase lag at a specific frequency. Furthermore, it can be useful to decompose a real signal into specific harmonic components of the spin speed. One such instance is in the analysis of a fluid film bearing. Often, the fluid film bearing will cause an unbalance at the subsynchronous frequency of just under 0.5X. Using a filter, the response of the system to this specific frequency can be extracted, allowing the analysis of phase angle and amplitude directly due to the influence of interest.

MATLAB has an extensive library of digital filters that can be adjusted to filter specific frequency ranges with no phase delay, and recall the states from the previous window as to not loose dynamic information from one step to the next. Explanation of filtering in MATLAB has been spared from this work as it is out of the scope.

A synchronous filter was applied the experimental example and the cascade plot is shown in Figure 1.9. All amplitudes of frequencies other than the synchronous frequencies have been eliminated. This system did not have strong super- or sub-synchronous response, so this filtering does not make an appreciable effect to the

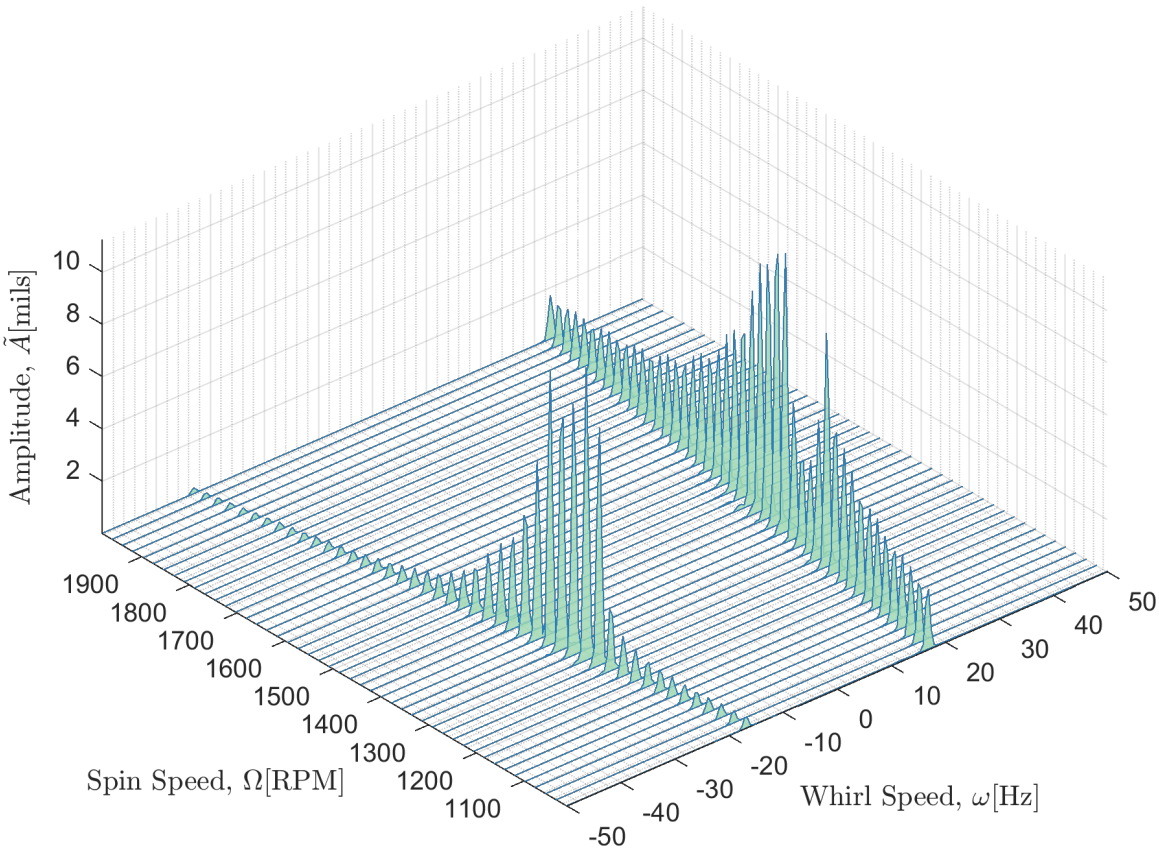


Figure 1.9: Cascade of the experimental system with a synchronous filter applied.

Bode plot. But, with many real systems filtering will be necessary to analyze the system.

1.2.5 Frequency and Time Resolution

Inherit in the application of the windowing method to the transient data set is some trade off with resolution in time and resolution in frequency content. As mentioned before, the frequency resolution, f_{res}, is equal to the sampling rate, f_s, divided by the number of samples per window, $nspw$. So given a specific sampling rate, frequency resolution is inversely proportional to the number of samples per window. But, $nspw$ is also inversely proportional to the number of windows in the signal, NW. The total length of time of the signals is then divided up into NW windows where. It is often

more convenient in rotordynamics to talk about change in speed instead of time, as it is more often the independent variable. Total speed range of Ω is divided up into NW windows, making the speed resolution (the difference in speed from one window to the next) proportional to NW. Therefore, as NW increases, f_{res} increases, and Ω_{res} decreases. Where Speed resolution is being referenced here by the variable Ω_{res}.

This relationship is easily explored through the use of the Cascade plot since it contains both frequency and speed information. An example of a high $nspw$, low NW, low f_{res}, & high Ω_{res} is found in Figure 1.10. The result of low f_{res}, at 0.2[Hz], is sharp peaks in whirl speed, but since the amplitude spectrum is changing with speed increase, the spectrum is spread over a large frequency range of $\Omega_{res} = 54[RPM]$. The opposite condition leading to low Ω_{res} and high f_{res} is presented in Figure 1.11. In this plot, the $f_{res} = 2[Hz]$ is so high that the whirl speeds are spread over a large range. On the other hand, the resulting Ω_{res} of 5.5[RPM] provides detailed information on the effect of changing spin speed. Also, with such high f_{res} a ripple effect can be seen in the Amplitude spectrum as the actual dominate whirl speed is between the resolution of 2[Hz] and bleeds into the nearest multiples of 2[Hz]. This effect is not representative of the actual amplitude spectrum of the signal and is avoided by choosing a lower f_{res}. The choice of f_{res} is dependent on many factors, including: f_s, spin speed ramp rate, and spin speed range.

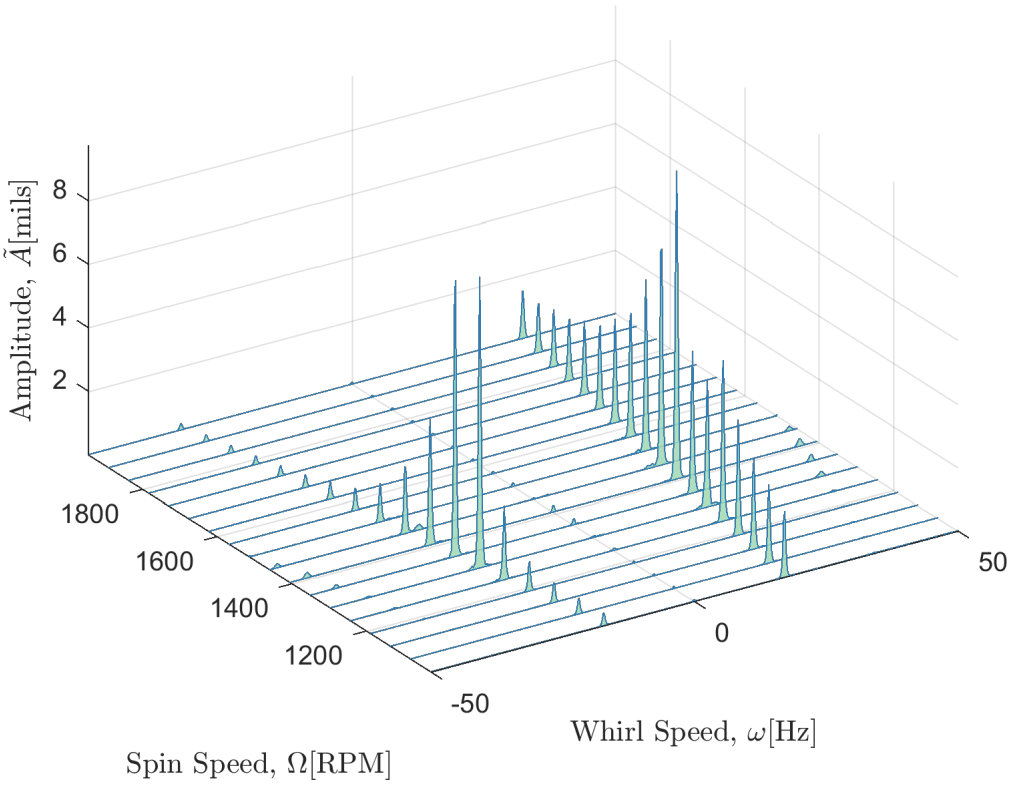


Figure 1.10: Cascade of the experimental system with an f_{res} of 0.2[Hz] and a resulting Ω_{res} of 54[RPM].

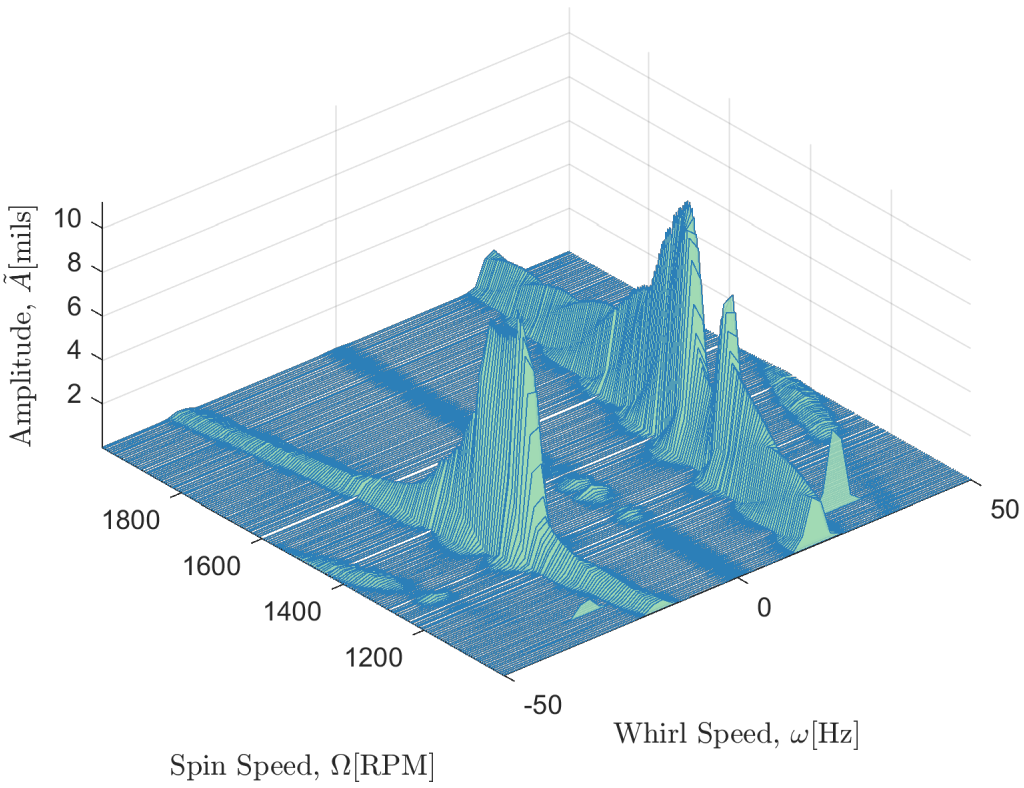


Figure 1.11: Cascade of the experimental system with an f_{res} of 2[Hz] and a resulting Ω_{res} of 5.5[RPM].

Chapter 2

FINITE ELEMENT METHOD FOR ROTORDYNAMIC SYSTEMS

In this chapter the Finite Element Method(FEM) will be employed in the simulation of rotordynamic systems. There are many advantages to using this discretized method to solve problems of rotating machines. One of which is the generic form that the global equation, and its parts take. The method makes it possible to define all components of a system separately, only to combine them at the end. Also, FEM makes it possible to move components around in space without a need to reevaluate the physics. Finally, the analysis techniques for the resulting equations of motion are wide-reaching, and will serve to richly enhance our understanding of a complex system made of simple parts.

First, the beam element commonly refered to as the Timoshenko Beam Element will be derived from the kinematic and constitutive constraints (derivation of an alternative element, the Bernoulli-Euler beam element is provided in Appendix A). The solution to the resulting equation of motion will be discretized and variables separated in position and time to give the finite element equations of motion. An extension to the Timoshenko beam element model will be presented that will consider viscous damping effects in the beam element. Equations for disks, bearings, and complex versions of all equations will be presented.

2.1 Timoshenko Beam Finite Element

The Timoshenko beam element allows for the beam cross section plane at any axis location to differ from normal with the axis of the beam. In other words, the element allows for shear stresses. This element often also includes the effects of rotary inertia

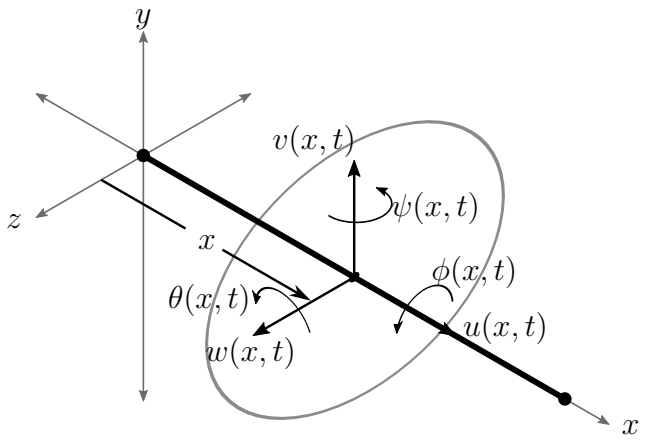


Figure 2.1: Timoshenko beam section with degrees of freedom at some point x along beam axis.

and gyroscopic moments, as it will in this derivation. Generalized displacements used are assumed to be variable in both time and space. The element has six degrees of freedom–three translation and three rotations all defined on the beam axis. Beam displacement coordinates are defined in Figure 2.1. All displacements are functions of time, t, and the axial spacial coordinate, x. Derivations herein are a synthesis from references provided in the literature review, further interpretations are adapted from: [1];[11];[16]. For internal damping interpretations and sources: [6];[15];[9];[7];[23].

2.1.1 Kinematic Relationships

In order to develop the relationship between internal stresses and strains in the beam, and subsequently the equations of motion, the motion of some arbitrary point on the beam must be defined in terms of the generalized coordinates. Motion of two points is taken into consideration to assist in dividing the motion into a translation and a rotation. These points are shown in Figure 2.2. The first point, C, falls on the beam axis at location x, and in the undeformed configuration, the vector $\vec{r}_c' = \overrightarrow{OC}$ forms a right angle with the surface of the cross section. The second point, P is at some arbitrary (y, z) location on the cross section. The vecor $\vec{t} = \overrightarrow{CP}$ points from the beam

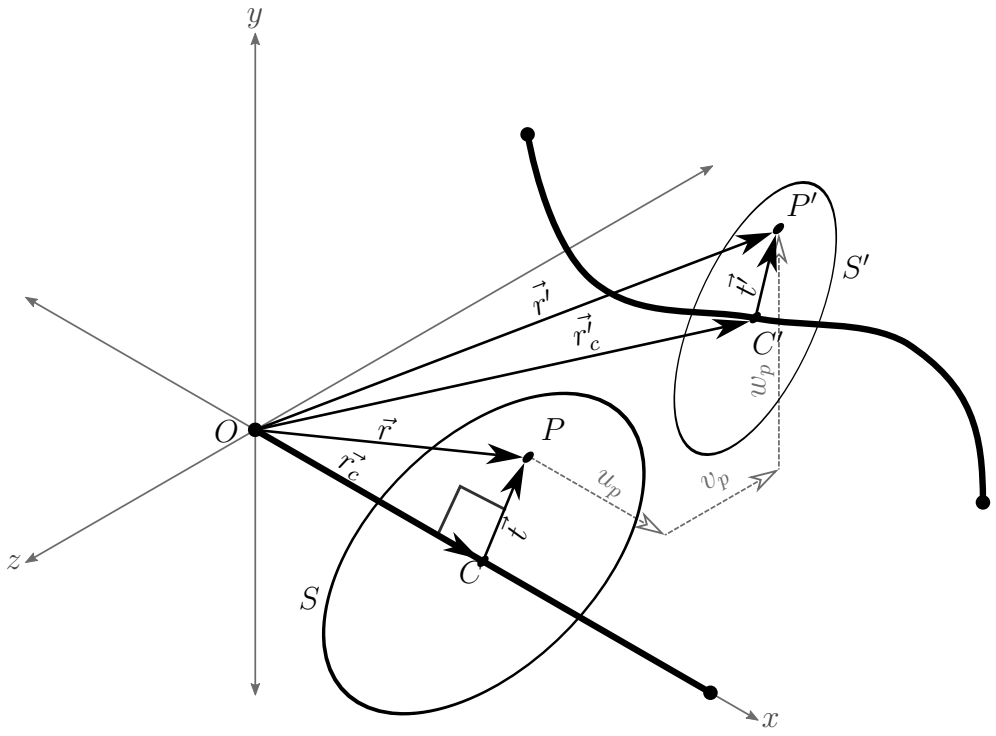


Figure 2.2: Beam Element with nodal displacements.

axis to the point along the cross section. If we follow this point P we will be able to define the motion of the cross section as a whole, or more succinctly, to define the displacements u_p, v_p, w_p in terms of the coordinates $u, v, w, \psi, \theta,$ & ϕ. The motion of point P is split into translation and rotation, where $\vec{t}$ is rotated and $\vec{r}_c$ is translated to point to the deformed location P'. The vector pointing to P in the undeformed state is defined as

$$\vec{r} = \vec{r}_c + \vec{t} \tag{2.1}$$

Rotations are represented with a rotation transformation matrix,

$$\vec{t'} = \underline{R}\vec{t} \tag{2.2}$$

The linearized first order rotational matrix for small angles is represented by

$$\underline{R} = \begin{bmatrix} 1 & -\theta & \psi \\ \theta & 1 & -\phi \\ -\psi & \phi & 1 \end{bmatrix} \tag{2.3}$$

and the translation with a displacement vector

$$\vec{r'_c} = \vec{r}_c + \vec{u} \tag{2.4}$$

Combined motion from P to P' can be defined by

$$\vec{u}_p = \vec{r'} - \vec{r} \tag{2.5}$$

where $\vec{u}_p$ is the vector containing u_p, v_p, & w_p. The vector definitions for $\vec{r'}$ and $\vec{r}$ are substituted to the above equation to obtain

$$\vec{u}_p = \vec{r'_c} + \vec{t'} - \vec{r}_c + \vec{t} \tag{2.6}$$

and now using the definition for $\vec{u}$ and $\vec{t'}$ leads to the simplified expression for the motion of P

$$\vec{u}_p = \vec{u} + (\underline{R} - \underline{I})\vec{t} \tag{2.7}$$

expanding matrices reveals the equation

$$\vec{u}_p = \begin{Bmatrix} u_p \\ v_p \\ w_p \end{Bmatrix} = \begin{Bmatrix} u \\ v \\ w \end{Bmatrix} + \begin{bmatrix} 0 & -\theta & \psi \\ \theta & 0 & -\phi \\ -\psi & \phi & 0 \end{bmatrix} \begin{Bmatrix} 0 \\ y \\ z \end{Bmatrix} \tag{2.8}$$

Therefore, the motion of any point on the beam may be approximated with

$$\vec{u}_p = \begin{Bmatrix} u - \theta y + \psi z \\ v - \phi z \\ w + \phi y \end{Bmatrix} \tag{2.9}$$

This equation will be used, in conjunction with material properties, to integrate over the beam cross sectional area and obtain generalized internal beam forces and moments.

2.1.2 Internal Constitutive Relationship

Stresses are assumed to exist in the beam in the axial direction, and in shear on the face of the beam section. Stresses in the transverse, or the y and z, directions are assumed negligible. Shear stresses out of the plane section are assumed to be vanishing as the differential element shrinks. Internal damping is to be considered independently from this material constitutive relationship. Written out, these stresses are represented by the matrix

$$\sigma_{ij} = \begin{bmatrix} \sigma_{xx} & \sigma_{xy} & \sigma_{xz} \\ \sigma_{xy} & 0 & 0 \\ \sigma_{xz} & 0 & 0 \end{bmatrix} \tag{2.10}$$

Using the Hooke's Law for a linear elastic isotropic material, expressed as

$$\epsilon_{ij} = \frac{1}{E}[(1+\nu)\sigma_{ij} - \nu\delta_{ij}\sigma_{kk}] \tag{2.11}$$

allows the determination of the stress strain relationship in engineering notation as

$$\begin{Bmatrix} \sigma_{xx} \\ \sigma_{xy} \\ \sigma_{xz} \end{Bmatrix} = \begin{bmatrix} E & 0 & 0 \\ 0 & 2G & 0 \\ 0 & 0 & 2G \end{bmatrix} \begin{Bmatrix} \epsilon_{xx} \\ \epsilon_{xy} \\ \epsilon_{xz} \end{Bmatrix} \tag{2.12}$$

where, $G = \frac{E}{2(1+\nu)}$. Strains are derived from displacements of equation (2.9) using a linear strain-displacement relationship for infinitesimal strains: $2\epsilon_{ij} = u_{i,j} + u_{j,i}$. Non-trivial internal strains are

$$\begin{cases} \epsilon_{xx} = u' - \theta' y + \psi' z \\ \epsilon_{xy} = \frac{1}{2}(v' - \phi' z - \theta) \\ \epsilon_{xz} = \frac{1}{2}(w' + \phi' y + \psi) \end{cases} \tag{2.13}$$

Note that in the case of the Euler-Bernoulli beam derivation the slope in a transverse direction displacement is equal to the rotation angle about the orthogonal transverse axis, i.e., $v' = \theta$ and $w' = \psi$. Application of those would reduce the system to the Euler-Bernoulli beam.

It will be proven useful to introduce generalized strains that group strain contributions above as axial, bending, torsion, and shear, represented by the symbols ε, ρ, φ, γ,respectively, as

$$\begin{cases} \varepsilon = u' \\ \rho_y = -\theta' \\ \rho_z = \psi' \\ \varphi = \phi' \\ \gamma_y = v' - \theta \\ \gamma_z = w' + \psi \end{cases} \tag{2.14}$$

Plugging in theses new generalized strains in (2.13) gives

$$\begin{cases} \epsilon_{xx} = \varepsilon + \rho_y y + \rho_z z \\ \epsilon_{xy} = \frac{1}{2}(\gamma_y - \varphi z) \\ \epsilon_{xz} = \frac{1}{2}(\gamma_z + \varphi y) \end{cases} \tag{2.15}$$

Now the stresses in equation (2.12) can be represented with the displacements or the generalized strains as

$$\left\{\begin{array}{c}\sigma_{xx}\\ \sigma_{xy}\\ \sigma_{xz}\end{array}\right\} = \left\{\begin{array}{c}E(u' - \theta' y + \psi' z)\\ G(v' - \phi' z - \theta)\\ G(w' + \phi' y + \psi)\end{array}\right\} = \left\{\begin{array}{c}E(\varepsilon + \rho_y y + \rho_z z)\\ G(\gamma_y - \varphi z)\\ G(\gamma_z + \varphi y)\end{array}\right\} \tag{2.16}$$

Total strain energy from internal stresses can be expressed by the integral expression

$$U = \int_V \sigma_{ij}\epsilon_{ij} dV \tag{2.17}$$

and expanded as

$$U = \int_V \left[\sigma_{xx}\epsilon_{xx} + 2\sigma_{xy}\epsilon_{xy} + 2\sigma_{xz}\epsilon_{xz}\right] dV \tag{2.18}$$

Expand strains using generalized strain expressions (2.15) and collect terms on generalized strains to get

$$U = \int_V \left[\sigma_{xx}\varepsilon + \sigma_{xx} y\rho_y + \sigma_{xx} z\rho_z + \sigma_{xy}\gamma_y + \sigma_{xz}\gamma_z + (\sigma_{xz} y - \sigma_{xy} z)\varphi\right] dV \tag{2.19}$$

This internal mechanical energy expression allows us to recognize stresses conjugate with each generalized strain as the corresponding stress for that phenomena. Integration allows the determination of the forces and moments related to each generalized strain as

$$\left\{\begin{array}{lll} N = & \int_A \sigma_{xx} dA & = E(A\varepsilon + S_y\rho_y + S_z\rho_z)\\ M_y = & \int_A \sigma_{xx} z dA & = E(S_z\varepsilon + I_{xy}\rho_y + I_y\rho_z)\\ M_z = & \int_A \sigma_{xx} y dA & = E(S_y\varepsilon + I_z\rho_y + I_{xy}\rho_z)\\ Q_y = & \int_A \sigma_{xy} dA & = \kappa G(A\gamma_y - S_z\varphi)\\ Q_z = & \int_A \sigma_{xz} dA & = \kappa G(A\gamma_z + S_y\varphi)\\ M_x = & \int_A (\sigma_{xz} y - \sigma_{xy} z) dA & = \kappa G\left(A_y\gamma_z - A_z\gamma_y + J_x\varphi\right)\end{array}\right. \tag{2.20}$$

$$\text{where, } \left\{\begin{array}{ll} \kappa = \frac{6(1+\nu)}{7+6\nu}, \textit{for circular cross sections.} & \\ A = \int_A dA & S_y = \int_A y dA\\ S_z = \int_A z dA & I_y = \int_A z^2 dA\\ I_z = \int_A y^2 dA & J_x = I_y + I_z\end{array}\right.$$

κ is the shear coefficient which attempts to correct for the fact that the shear strain is not constant over the beam cross section. Assuming the central axis of the beam is coincident with the shear center, then $A_y = A_z = I_{xy} = 0$. Which simplifies the conjugate forces to

$$\begin{cases} N = EA\varepsilon & = EAu' \\ M_y = EI_y\rho_z & = EI_y\psi' \\ M_z = EI_z\rho_y & = -EI_z\theta' \\ Q_y = \kappa GA\gamma_y & = \kappa GA(v' - \theta) \\ Q_z = \kappa GA\gamma_z & = \kappa GA(w' + \psi) \\ M_x = \kappa GJ_x\varphi & = \kappa GJ_x\phi' \end{cases} \tag{2.21}$$

These forces will be used to evaluate the motion of a beam element using a free body diagram and resultant motion.

2.1.3 Differential Equations of Motion

The equations of motion will now be derived for the Timoshenko beam element. External forces are not included in this derivation. Though they may easily be added to the diagram of figure 2.3 and included in the analysis. It is also assumed that the cross section remains planar during deformation and the material properties are homogeneous through time and space. The derivation is the same as for a Euler-Bernoulli beam with the exception of the constitutive relations used at the end and the inclusion of torsion and axial degrees of freedom. Using conservation of momentum and conservation of the moment of momentum a relationship between inertia and internal forces is developed. Using Figure 2.3 and applying summation of forces in the x-direction leads to

$$(N + \frac{1}{2}\frac{\partial N}{\partial x}dx) - (N - \frac{1}{2}\frac{\partial N}{\partial x}dx) = \rho A dx \frac{\partial^2 u}{\partial x^2} \tag{2.22}$$

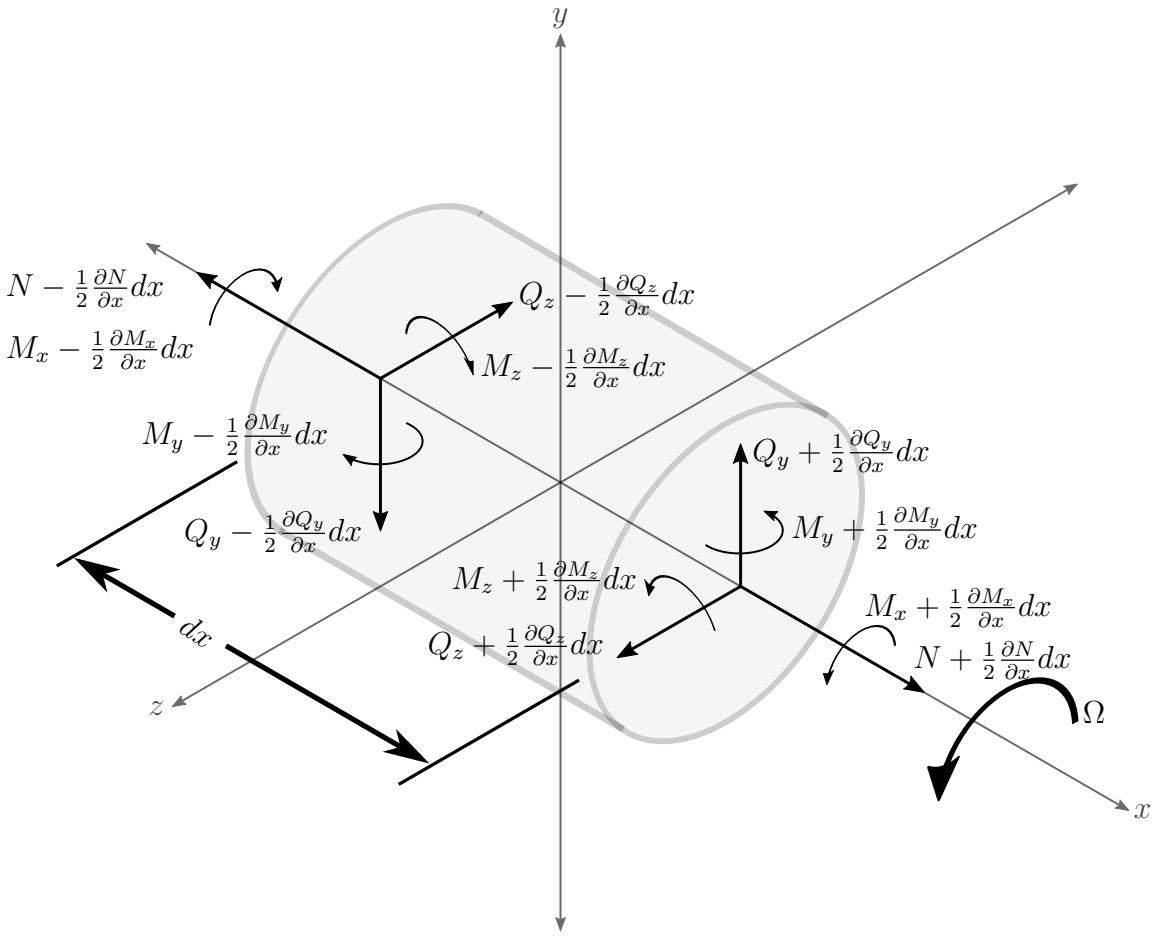


Figure 2.3: Beam differential element with generalized forces.

By Simplifying the above equation, and performing the same steps for the other directions and moments we get

$$\begin{cases} \frac{\partial N}{\partial x} = \rho A \frac{\partial^2 u}{\partial x^2} \\ \frac{\partial Q_y}{\partial x} = \rho A \frac{\partial^2 v}{\partial x^2} \\ \frac{\partial Q_z}{\partial x} = \rho A \frac{\partial^2 w}{\partial x^2} \\ \frac{\partial M_y}{\partial x} - Q_z = \rho I_y \frac{\partial^2 \psi}{\partial x^2} + \rho J_x \Omega \frac{\partial \theta}{\partial x} \\ \frac{\partial M_z}{\partial x} + Q_y = \rho I_z \frac{\partial^2 \theta}{\partial x^2} - \rho J_x \Omega \frac{\partial \psi}{\partial x} \\ \frac{\partial M_x}{\partial x} = \rho J_x \frac{\partial^2 \phi}{\partial x^2} \end{cases} \tag{2.23}$$

Gyroscopic moments have been explicitly added to the summations in the appropriate equations. The generalized forces of equation (2.21) are substituted in the equilibrium equations (2.23)

$$\begin{cases} EAu'' = \rho A \ddot{u} & \text{(2.24a)} \\ \kappa GA(v'' - \theta') = \rho A \ddot{v} & \text{(2.24b)} \\ \kappa GA(w'' + \psi') = \rho A \ddot{w} & \text{(2.24c)} \\ EI_y \psi'' - \kappa GA(w' + \psi) = \rho I_y \ddot{\psi} + \rho J_x \Omega \dot{\theta} & \text{(2.24d)} \\ EI_z \theta'' + \kappa GA(v' - \theta) = \rho I_z \ddot{\theta} - \rho J_x \Omega \dot{\psi} & \text{(2.24e)} \\ \kappa GJ_x \phi'' = \rho J_x \ddot{\phi} & \text{(2.24f)} \end{cases}$$

In matrix form, this system of equations can be represented by this equation

$$\underline{\mathcal{M}}^e \ddot{\vec{\mathbf{u}}} + \Omega \underline{\mathcal{G}}^e \dot{\vec{\mathbf{u}}} - \left(\frac{\partial()}{\partial x} \underline{\mathcal{S}}^e - \underline{\mathcal{P}}^e \underline{\mathcal{S}}^e \right) \vec{\mathbf{u}} = 0 \tag{2.25}$$

where,

$$\begin{cases} \underline{\mathcal{M}}^e = \begin{bmatrix} \rho A & 0 & 0 & 0 & 0 & 0 \\ 0 & \rho A & 0 & 0 & 0 & 0 \\ 0 & 0 & \rho I_y & 0 & 0 & 0 \\ 0 & 0 & 0 & \rho I_z & 0 & 0 \\ 0 & 0 & 0 & 0 & \rho A & 0 \\ 0 & 0 & 0 & 0 & 0 & \rho J_x \end{bmatrix} \quad \underline{\mathcal{G}}^e = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \rho J_x & 0 & 0 \\ 0 & 0 & -\rho J_x & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \\ \underline{\mathcal{P}}^e = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad \underline{\mathcal{S}}^e = \begin{bmatrix} \kappa GA\frac{\partial()}{\partial x} & 0 & 0 & -\kappa GA & 0 & 0 \\ 0 & \kappa GA\frac{\partial()}{\partial x} & \kappa GA & 0 & 0 & 0 \\ 0 & 0 & EI_y\frac{\partial()}{\partial x} & 0 & 0 & 0 \\ 0 & 0 & 0 & EI_z\frac{\partial()}{\partial x} & 0 & 0 \\ 0 & 0 & 0 & 0 & EA\frac{\partial()}{\partial x} & 0 \\ 0 & 0 & 0 & 0 & 0 & \kappa GJ_x\frac{\partial()}{\partial x} \end{bmatrix} \end{cases} \tag{2.26}$$

and $\vec{\mathbf{u}} = [v, w, \psi, \theta, u, \phi]^\intercal$. The principle of virtual displacements is utilized on the equations of motion to obtain the weak form of the equations of motion and integrated over the length of the beam.

$$\int_0^l \delta\vec{\mathbf{u}}^\intercal \underline{\mathcal{M}}^e \ddot{\vec{\mathbf{u}}} dx + \int_0^l \delta\vec{\mathbf{u}}^\intercal \Omega \underline{\mathcal{G}}^e \dot{\vec{\mathbf{u}}} - \int_0^l \delta\vec{\mathbf{u}}^\intercal \frac{\partial()}{\partial x} \underline{\mathcal{S}}^e \vec{\mathbf{u}} dx + \int_0^l \delta\vec{\mathbf{u}}^\intercal \underline{\mathcal{P}} \underline{\mathcal{S}}^e \vec{\mathbf{u}} dx = 0 \tag{2.27}$$

integration by parts on the third term and replacing $\underline{\mathcal{S}}$ with $\underline{\mathcal{D}}^e\underline{\mathcal{B}}$, and making use of the Identity matrix, $\underline{\mathbf{I}}$ where $\frac{\partial()}{\partial x}\underline{\mathbf{I}}$ is interpreted here as if the partial was a scalar

$$\int_0^l \delta\vec{\mathbf{u}}^\intercal \underline{\mathcal{M}}^e \ddot{\vec{\mathbf{u}}} dx + \Omega \int_0^l \delta\vec{\mathbf{u}}^\intercal \underline{\mathcal{G}}^e \dot{\vec{\mathbf{u}}} + \int_0^l \delta\vec{\mathbf{u}}^\intercal \left(\frac{\partial()}{\partial x}\underline{\mathbf{I}} + \underline{\mathcal{P}} \right) \underline{\mathcal{D}}^e \underline{\mathcal{B}} \vec{\mathbf{u}} dx = 0 \tag{2.28}$$

$$\underline{\mathcal{D}}^e = \begin{bmatrix} \kappa GA & 0 & 0 & 0 & 0 & 0 \\ 0 & \kappa GA & 0 & 0 & 0 & 0 \\ 0 & 0 & EI_y & 0 & 0 & 0 \\ 0 & 0 & 0 & EI_z & 0 & 0 \\ 0 & 0 & 0 & 0 & EA & 0 \\ 0 & 0 & 0 & 0 & 0 & \kappa GJ_x \end{bmatrix} \quad \& \quad \underline{\mathcal{B}} = \begin{bmatrix} \frac{\partial()}{\partial x} & 0 & 0 & -1 & 0 & 0 \\ 0 & \frac{\partial()}{\partial x} & 1 & 0 & 0 & 0 \\ 0 & 0 & \frac{\partial()}{\partial x} & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{\partial()}{\partial x} & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{\partial()}{\partial x} & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{\partial()}{\partial x} \end{bmatrix} \tag{2.29}$$

Notice that $\frac{\partial()}{\partial x}\underline{I} + \underline{\mathcal{P}} = \underline{\mathcal{B}}^\intercal$ so the equation of motion becomes

$$\int_0^l \delta\vec{\mathbf{u}}^\intercal \underline{\mathcal{M}}^e \ddot{\vec{\mathbf{u}}} dx + \Omega \int_0^l \delta\vec{\mathbf{u}}^\intercal \underline{\mathcal{G}}^e \dot{\vec{\mathbf{u}}} + \int_0^l \delta\vec{\mathbf{u}}^\intercal \underline{\mathcal{B}}^\intercal \underline{\mathcal{D}}^e \underline{\mathcal{B}} \vec{\mathbf{u}} dx = 0 \tag{2.30}$$

$\underline{\mathcal{M}}^e$ is the inertia of the element, $\underline{\mathcal{G}}^e$ is the rotating inertia, $\underline{\mathcal{D}}^e$ is the material stress-strain relationship, and $\underline{\mathcal{B}}^e$ is the strain-displacement operator. The solution of this differential system motivates a separation of variables that will be discussed in the next section.

2.1.4 Shape Functions

The displacements thus far have been assumed to be functions of both position and time. Now the total displacement is separated into functions that depend on time and functions that depend on position. This is a fundamental part of the discretization of the beam element, and the use the finite element method.

$$\begin{cases} \vec{\mathbf{u}}(x,t) &= \underline{\mathbf{N}}(x)\vec{\mathbf{q}}(t) \\ \dot{\vec{\mathbf{u}}}(x,t) &= \underline{\mathbf{N}}(x)\dot{\vec{\mathbf{q}}}(t) \\ \ddot{\vec{\mathbf{u}}}(x,t) &= \underline{\mathbf{N}}(x)\ddot{\vec{\mathbf{q}}}(t) \\ \delta\vec{\mathbf{u}}(x,t) &= \underline{\mathbf{N}}(x)\delta\vec{\mathbf{q}}(t) \end{cases} \tag{2.31}$$

$\vec{\mathbf{u}} = [v, w, -\psi, \theta, u, \phi]^\intercal$ & $\vec{\mathbf{q}} = [v_1, w_1, -\psi_1, \theta_1, v_2, w_2, -\psi_2, \theta_2, u_1, \phi_1, u_2, \phi_2]^\intercal$. This specfic order of $\vec{\mathbf{q}}$ is chosen with u and ϕ at the end to ease the condensation of the axial and torsional degrees of freedom out of the system if their use is not necessary for the system of interest. ψ angles are defined as negative to allow for the same stiffness matrix to define the motion in both planes, and more importantly, to allow for the use of the complex plane to simplify the problem. The shape functions $\underline{\mathbf{N}}(x)$ interpolate the displacements between the beam ends. These functions must must solve the static portion of the differential equations (2.24). These shape functions are chosen as a polynomials that satisfy the boundary nodal displacements and rotations at the ends of a beam element. These nodal degrees of freedom, depicted in Figure 2.4 are considered to be interpolated through the beam element by the shape functions. Interpolation functions chosen are listed in Equation (2.32). Axial dis-

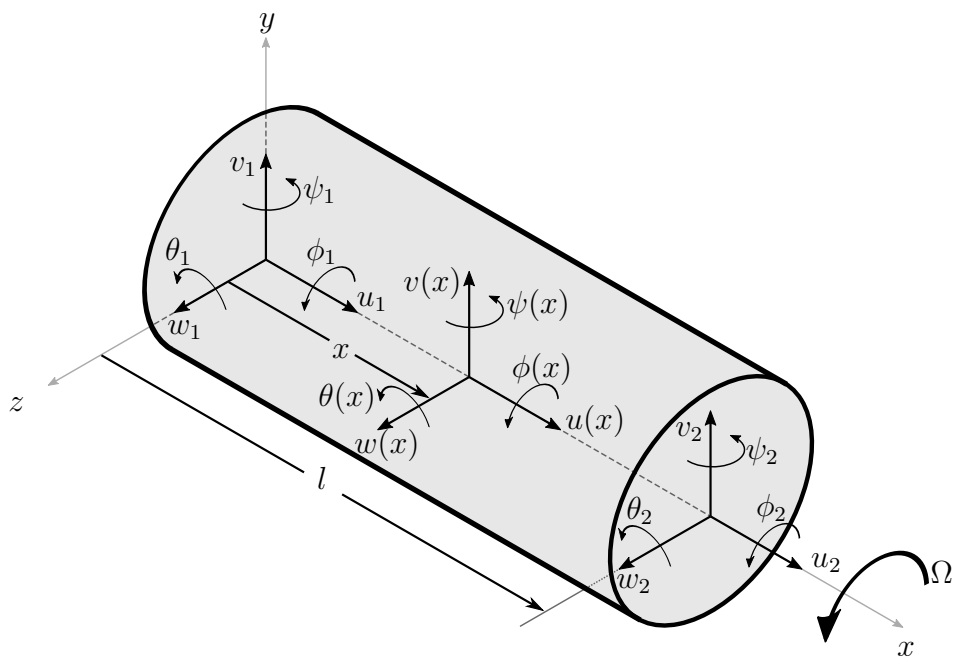


Figure 2.4: Beam Element with nodal displacements.

placement, u, and torsional rotation, ϕ are independent, so their shape functions are chosen as polynomials that satisfy the differential equation. Conversely, transverse displacements, v & w, and bending rotations, ψ & θ are coupled. Coupling of the shape functions has been proven to reduce some negative effects of linearly interpolated elements [17]. Polynomial functions are chosen for v & w and their rotational counterparts are derived using the differential relations.

$$\begin{cases} u = c_1 + c_2 x & \text{(2.32a)} \\ v = c_3 + c_4 x + c_5 x^2 + c_6 x^3 & \text{(2.32b)} \\ w = c_7 + c_8 x + c_9 x^2 + c_{10} x^3 & \text{(2.32c)} \\ \phi = c_{11} + c_{12} x & \text{(2.32d)} \end{cases}$$

$c_{1,2,\ldots}$ are the unknown constants of the polynomial solutions. Using transverse displacement of equations (2.32b) & (2.32c) in the differential equations (2.24b), (2.24c),

(2.24d), (2.24e) the interpolation functions of bending rotations are derived as:

$$\begin{cases} \psi = K_y c_{10} - c_8 - 2c_9 x - 3c_{10} x^2 & \text{(2.33a)} \\ \theta = K_z c_6 + c_4 + 2c_5 x + 3c_6 x^2 & \text{(2.33b)} \end{cases}$$

where, $K_y = \frac{6EI_y}{\kappa GA}$& $K_z = \frac{6EI_z}{\kappa GA}$.

Boundary conditions for the interpolation polynomials of equations (2.32) & (2.33) are defined as the components of the vector $\vec{\mathbf{q}} u_j = u(x_j)$ and similarily for other degrees of freedom. Where, $j = 1, 2$ and defines the two states. In this derivation, $x_1 = 0$ and $x_2 = l$. Application of these boundary condition results in the below relationship with the polynomial constants.

$$\begin{Bmatrix} u_1 \\ u_2 \\ v_1 \\ v_2 \\ w_1 \\ w_2 \\ \psi_1 \\ \psi_2 \\ \theta_1 \\ \theta_2 \\ \phi_1 \\ \phi_2 \end{Bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & l & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & l & l^2 & l^3 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & l & l^2 & l^3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & -K_y & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & -2l & -K_y - 3l^2 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & K_z & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 2l & K_z + 3l^2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & l \end{bmatrix} \begin{Bmatrix} c_1 \\ c_2 \\ c_3 \\ c_4 \\ c_5 \\ c_6 \\ c_7 \\ c_8 \\ c_9 \\ c_{10} \\ c_{11} \\ c_{12} \end{Bmatrix} \quad \text{(2.34)}$$

Inversion of this matrix results in a system of equations defining the constant c_1 through c_{12}. These constants are then substituted in to the polynomial expressions

(2.32) & (2.33) giving the interpolations as functions of the nodal displacements as

$$\begin{cases} u = N_1 u_1 + N_2 u_2 \\ v = T_{t_1 y} v_1 + T_{t_2 y} v_2 + T_{r_1 y} \theta_1 + T_{r_2 y} \theta_2 \\ w = T_{t_1 z} w_1 + T_{t_2 w} w_2 + T_{r_1 z} \psi_1 + T_{r_2 z} \psi_2 \\ \psi = R_{t_1 z} w_1 + R_{t_2 w} w_2 + R_{r_1 z} \psi_1 + R_{r_2 z} \psi_2 \\ \theta = R_{t_1 y} v_1 + R_{t_2 y} v_2 + R_{r_1 y} \theta_1 + R_{r_2 y} \theta_2 \\ \phi = N_1 \phi_1 + N_2 \phi_2 \end{cases} \tag{2.35}$$

with,

$$\begin{cases} N_1 = 1 - \zeta & N_2 = \zeta \\ T_{t_1 y,z} = \frac{1}{1+\alpha_{y,z}}(2\zeta^3 - 3\zeta^2 - \alpha_{y,z}\zeta + 1 + \alpha_{y,z}) & T_{t_2 y,z} = \frac{1}{1+\alpha_{y,z}}(-2\zeta^3 + 3\zeta^2 + \alpha_{y,z}\zeta) \\ T_{r_1 y,z} = \frac{l}{1+\alpha_{y,z}}[\zeta^3 - (2 + \frac{1}{2}\alpha_{y,z})\zeta^2 + (1 + \frac{1}{2}\alpha_{y,z})\zeta] & T_{r_2 y,z} = \frac{l}{1+\alpha_{y,z}}[\zeta^3 - (1 - \frac{1}{2}\alpha_{y,z})\zeta^2 - \frac{1}{2}\alpha_{y,z}\zeta] \\ R_{t_1 y,z} = \frac{6/l}{1+\alpha_{y,z}}(\zeta^2 - \zeta) & R_{t_2 y,z} = \frac{6/l}{1+\alpha_{y,z}}(-\zeta^2 + \zeta) \\ R_{r_1 y,z} = \frac{1}{1+\alpha_{y,z}}(3\zeta^2 - (4 + \alpha_{y,z})\zeta + 1 + \alpha_{y,z}) & R_{r_2 y,z} = \frac{1}{1+\alpha_{y,z}}(3\zeta^2 - (2 - \alpha_{y,z})\zeta) \end{cases} \tag{2.36}$$

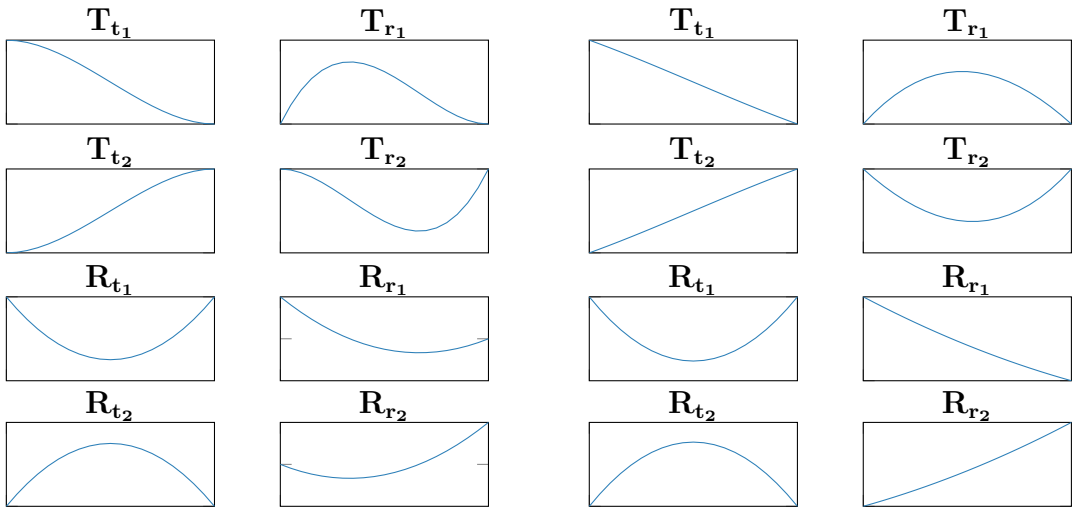
where, $\alpha_y = 2K_y/l^2 = \frac{12EI_y}{\kappa GAl^2}$, $\alpha_z = 2K_z/l^2 = \frac{12EI_z}{\kappa GAl^2}$, & $\zeta = x/l$. (2.35) is expressed in matrix form as it appears in (2.31) where

$$\underline{\mathbf{N}}(x) = \begin{bmatrix} T_{t_1 y} & 0 & 0 & T_{r_1 y} & T_{t_2 y} & 0 & 0 & T_{r_2 y} & 0 & 0 & 0 & 0 \\ 0 & T_{t_1 z} & T_{r_1 z} & 0 & 0 & T_{t_2 z} & T_{r_2 z} & 0 & 0 & 0 & 0 & 0 \\ 0 & R_{t_1 z} & R_{r_1 z} & 0 & 0 & R_{t_2 z} & R_{r_2 z} & 0 & 0 & 0 & 0 & 0 \\ R_{t_1 y} & 0 & 0 & R_{r_1 y} & R_{t_2 y} & 0 & 0 & R_{r_2 y} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & N_1 & 0 & N_2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & N_1 & 0 & N_2 \end{bmatrix} \tag{2.37}$$

still with the generalized displacement vector $\vec{\mathbf{q}} = [v_1, w_1, -\psi_1, \theta_1, v_2, w_2, -\psi_2, \theta_2, u_1, \phi_1, u_2, \phi_2]^\intercal$

Shape functions depend on the term α which is sometimes called the shear correction factor. This shear correction factor is proportional to the square of the ratio of radius to length of the beam element. So, as the length increases relative to the radius, α tends to zero. It will be evident in the following section that as α approaches zero, the equations of motion approach the equations of the Bernoulli-Euler

beam. A spatial representation of the shape functions of equation (2.36) is given in figure 2.5. Shape functions plotted with respect to the non-dimensional length lend



(a) length to radius ratio of 100.

(b) length to radius ratio of 1.

Figure 2.5: Shape Functions as they vary with ζ using two different ratios of length to radius of beam element.

a visualization to the contribution of each shape function. Shape functions for axial and torsion are omitted since they are just linear polynomials. Each individual plot can be interpreted as a transformation from the corresponding input coordinate to the output. For instance, the first shape function plot is the output of $v(x)$ with an input of v_1 while all other coordinates are zero. The shape makes sense under this interpretation, as the translation starts at some value, v_1 and decreases to zero at the end since v_2 is zero. Also, notice how bending is nearly completely eliminated as the radius approaches the length as in Figure 2.5b. All shapes the Timoshenko beam will make in the model results is a linear combination of the shapes shown here.

2.1.5 Finite Equations of Motion

To obtain the equations of motion in terms of the generalized coordinates, $\vec{\mathbf{q}}$, displacement variables $\vec{\mathbf{u}}$ are replaced with definitions in equation 2.31.

$$\int_0^l \underline{\mathbf{N}}^{\intercal}\delta\vec{\mathbf{q}}^{\intercal}\underline{\mathcal{M}}^e\underline{\mathbf{N}}\ddot{\vec{\mathbf{q}}}dx + \Omega\int_0^l \underline{\mathbf{N}}^{\intercal}\delta\vec{\mathbf{q}}^{\intercal}\underline{\mathcal{G}}^e\underline{\mathbf{N}}\dot{\vec{\mathbf{q}}}dx + \int_0^l \underline{\mathbf{N}}^{\intercal}\delta\vec{\mathbf{q}}^{\intercal}\underline{\mathcal{B}}^{\intercal}\underline{\mathcal{D}}^e\underline{\mathcal{B}}\underline{\mathbf{N}}\vec{\mathbf{q}}dx = 0 \tag{2.38}$$

Note that $\vec{\mathbf{q}}$ is not dependent on x so it, and it's derivatives, may be pulled out of the integrals. Define $\underline{\mathbf{B}} = \underline{\mathcal{B}}\underline{\mathbf{N}}$ and substitute in, noting that $\underline{\mathbf{B}}$ interpolates strains from discrete displacements $\vec{\mathbf{q}}$. The motion equations are then

$$\int_0^l \underline{\mathbf{N}}^{\intercal}\underline{\mathcal{M}}^e\underline{\mathbf{N}}dx\ddot{\vec{\mathbf{q}}} + \Omega\int_0^l \underline{\mathbf{N}}^{\intercal}\underline{\mathcal{G}}^e\underline{\mathbf{N}}dx\dot{\vec{\mathbf{q}}} + \int_0^l \underline{\mathbf{B}}^{\intercal}\underline{\mathcal{D}}^e\underline{\mathbf{B}}dx\vec{\mathbf{q}} = 0 \tag{2.39}$$

Define

$$\begin{cases} \underline{\mathbf{M}}^e = \displaystyle\int_0^l \underline{\mathbf{N}}^{\intercal}\underline{\mathcal{M}}^e\underline{\mathbf{N}}dx & \text{(2.40a)} \\ \underline{\mathbf{G}}^e = \displaystyle\int_0^l \underline{\mathbf{N}}^{\intercal}\underline{\mathcal{G}}^e\underline{\mathbf{N}}dx & \text{(2.40b)} \\ \underline{\mathbf{K}}^e = \displaystyle\int_0^l \underline{\mathbf{B}}^{\intercal}\underline{\mathcal{D}}^e\underline{\mathbf{B}}dx & \text{(2.40c)} \end{cases}$$

so that, the general equations of motion for the timoshenko beam element are

$$\underline{\mathbf{M}}^e\ddot{\vec{\mathbf{q}}} + \Omega\underline{\mathbf{G}}^e\dot{\vec{\mathbf{q}}} + \underline{\mathbf{K}}^e\vec{\mathbf{q}} = 0 \tag{2.41}$$

notwithstanding the inclusion of viscous and hysteretic internal damping phenomena. Derivations, and inclusion of these phenomena in the equations of motion are to be included in the following section.

2.1.6 Rotating Internal Damping

Rotating damping is the main cause of instability in rotating machines. Non-rotating damping, such as the damping contributions from bearing supports, introduce a stabilizing effect. But, as rotating damping is dependent on rotation, its direction of

force can contribute to destabilization. Typically, friction components such as bearings with shrink fits or oil bearings are responsible for this destabilizing force. Due to the inherent complexity of modeling loose bearing components, or shrink fit dynamics, the analysis of the internal damping in the shaft elements is considered alone for this work. This will allow for the study of the destabilizing effect in general without requiring too much specificity in design parameters. Inclusion of the rotating damping effect in this way will allow for the design of components, and geometry of rotor system, to maximize stability. This stability analysis is only possible with the inclusion of some destabilizing force, [9],[10],[15],[24].

To motivate understanding of this force a simple derivation is provided with a rotating damping whose force is proportional to the flex of rate of change of the flex in the shaft. This is easier to define in the rotating reference frame, as the variables in this coordinate system directly represent the shaft flex from its neutral position. The rotating coordinates are located in the same plane as the fixed coordinates, but they rotate with the shaft speed, Ω. They are angularly displaced from the fixed coordinates at any point in time by the angle Ωt. The viscous damping in this rotating coordinate system is defined as

$$\vec{\mathcal{F}}_{\xi\eta} = -c_r \begin{Bmatrix} \dot{\xi}_c \\ \dot{\eta}_c \end{Bmatrix} \tag{2.42}$$

translating this force back to the stationary reference frame requires the rotation transformation matrix

$$\underline{\mathcal{R}} = \begin{bmatrix} \cos\Omega t & \sin\Omega t \\ -\sin\Omega t & \cos\Omega t \end{bmatrix} \tag{2.43}$$

which transforms stationary into rotating coordinates as

$$\begin{cases} \begin{Bmatrix} \xi_c \\ \eta_c \end{Bmatrix} &= \underline{\mathcal{R}} \begin{Bmatrix} y_c \\ z_c \end{Bmatrix} \\ \begin{Bmatrix} \dot{\xi}_c \\ \dot{\eta}_c \end{Bmatrix} &= \underline{\mathcal{R}} \begin{Bmatrix} \dot{y}_c \\ \dot{z}_c \end{Bmatrix} + \dot{\underline{\mathcal{R}}} \begin{Bmatrix} y_c \\ z_c \end{Bmatrix} \end{cases} \tag{2.44}$$

where,

$$\dot{\underline{\mathcal{R}}} = \Omega \begin{bmatrix} -\sin\Omega t & \cos\Omega t \\ -\cos\Omega t & -\sin\Omega t \end{bmatrix} \tag{2.45}$$

Substituting the second equation in 2.44 for the velocities in 2.42

$$\vec{\mathcal{F}}_{xy} = -c_r \begin{Bmatrix} \dot{y}_c \\ \dot{z}_c \end{Bmatrix} - c_r\Omega \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \begin{Bmatrix} y_c \\ z_c \end{Bmatrix} \tag{2.46}$$

From equation (2.46) a dependence on both velocity and position is evident. The portion dependent on the velocity is inherently stable as pulls opposite the motion. Conversely, the portion dependent on position cross couples the two displacements. This causes a destabilizing effect that grows as Ω increases. A net destabilizing force is produced once the latter portion of the exceeds the former. Without the presence of other non-rotating damping forces, the system will destabilize.

For the beam element, the constitutive relationship given in [24] is comprised of both viscous and hysteretic forms of damping, η_v & η_h respectively.

$$\sigma_{xx} = E\left\{ \frac{\epsilon_{xx}}{\sqrt{1+\eta_h^2}} + \left(\eta_v + \frac{\eta_h}{\omega\sqrt{1+\eta_h^2}} \right) \dot{\epsilon}_{xx} \right\} \tag{2.47}$$

Through the use of kinematics to obtain strain-displacement relations from the above constraints,

$$\begin{cases} \epsilon_{xx} &= -r\cos\left(\Omega-\omega\right) t \frac{\partial^2 R}{\partial x^2} \\ \dot{\epsilon}_{xx} &= (\Omega-\omega) r\sin\left(\Omega-\omega\right) t \frac{\partial^2 R}{\partial x^2} - r\cos\left(\Omega-\omega\right) t \frac{\partial}{\partial t}\frac{\partial^2 R}{\partial x^2} \end{cases} \tag{2.48}$$

and inspection to obtain moment equations,

$$\begin{cases} M_y &= \int_0^{2\pi}\int_0^a [w + r\sin\Omega t]\sigma_{xx} dr(rd(\Omega t)) \\ M_z &= \int_0^{2\pi}\int_0^a -[v + r\cos\Omega t]\sigma_{xx} dr(rd(\Omega t)) \end{cases} \tag{2.49}$$

we can complete a moment bending relationship to be used in the equations of motion

$$\begin{Bmatrix} M_y \\ M_z \end{Bmatrix} = EI \begin{bmatrix} \eta_a & \Omega\eta_v + \eta_b \\ \Omega\eta_v + \eta_b & -\eta_a \end{bmatrix} \begin{Bmatrix} v'' \\ w'' \end{Bmatrix} + EI \begin{bmatrix} \eta_v & 0 \\ 0 & -\eta_v \end{bmatrix} \begin{Bmatrix} \dot{v}'' \\ \dot{w}'' \end{Bmatrix} \tag{2.50}$$

where, $\eta_a = \frac{1+\eta_h}{\sqrt{1+\eta_h^2}}$ & $\eta_b = \frac{\eta_h}{\sqrt{1+\eta_h^2}}$

Using the same strategy followed when solving for the weak form of the beam differential equations following the Principle of Virtual Displacements starting at equation (2.27) will reveal the new equations of motion. Then use of a seperation of variables as defined by equation (2.31) to arrive at the total beam element equations of motion including internal damping as

$$\underline{\mathbf{M}}^e\ddot{\vec{\mathbf{q}}} + (\eta_v\underline{\mathbf{K}}^e + \Omega\underline{\mathbf{G}}^e)\dot{\vec{\mathbf{q}}} + [\eta_a\underline{\mathbf{K}}^e + (\Omega\eta_v + \eta_b)\underline{\mathbf{C}}^e]\vec{\mathbf{q}} = 0 \tag{2.51}$$

where,

$$\underline{\mathcal{I}} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad \& \quad \underline{\mathbf{C}}^e = \int_0^l \underline{\mathbf{B}}^\intercal \underline{\mathcal{I}}\underline{\mathcal{D}}\underline{\mathbf{B}} dx \tag{2.52}$$

$\underline{\mathbf{C}}^e$ is the "Circulation matrix", or the skew symmetric stiffness matrix.

2.1.7 Beam Element in Complex Coordinates

Complex coordinates collapse the equations of each plane into one set of equations. This lends properties to a axisymmetric rotor systems that will be exploited in the analysis of the model. For the complex analysis in this body of work, the system is assumed to be axisymmetric and the torsional and axial degrees of freedom are

omitted. Since the contributions from axial and torsional degrees of freedom are uncoupled from the system, there condensation has no effect on the remainder of system matrices. Complex coordinates used here are defined as:

$$\vec{\mathbf{s}} = \left\{ \begin{array}{c} \vec{r} \\ \vec{p} \end{array} \right\} = \left\{ \begin{array}{c} v + iw \\ \theta - i\psi \end{array} \right\} \tag{2.53}$$

Because the element is axisymmetric, and the special form of coordinates is used, symmetric beam equations in one plane hold for the complex plane and skew symmetric matrices become complex versions of the same matrix. Elemental matrices can be formed using the collapsed version of the shape functions matrix

$$\underline{\mathbf{N}}^c = \begin{bmatrix} T_{t_1 y} & T_{r_1 y} & T_{t_2 y} & T_{r_2 y} \\ R_{t_1} & R_{r_1} & R_{t_2} & R_{r_2} \end{bmatrix} \tag{2.54}$$

in

$$\vec{\mathbf{s}} = \underline{\mathbf{N}}^c \vec{\mathbf{q}}^c \tag{2.55}$$

where $\vec{\mathbf{q}}^c = [\vec{\mathbf{s}}_1, \vec{\mathbf{s}}_2]^\intercal$.

To convince this property of the system in the complex plane, a proof for the elemental equations of motion will be made. Taking equation (2.24b), adding equation (2.24c) multiplied by the imaginary unit i and using the definition for complex variables $\vec{p}$ and $\vec{r}$ leads to the transverse equation of motion in the complex plane

$$\kappa GA(\vec{r}'' - \vec{p}') = \rho A \ddot{\vec{r}} \tag{2.56}$$

and taking equation (2.24e) and subtracting equation (2.24d) multiplied by i gives the rotation equation in the complex plane

$$EI\vec{p}'' - \kappa GA(\vec{r}' - \vec{p}) = \rho I \ddot{\vec{p}} + i\rho J\Omega\dot{\vec{p}} \tag{2.57}$$

which by inspection of both of these equations it is evident that the form is the same as equations (2.24b) and (2.24e) except for the imaginary unit on the cross coupled

parts of the equation. Now that this idea is motivated, the finite element equations of motion analogous to equation (2.51) are

$$\underline{\mathbf{M}}^{ec}\ddot{\vec{\mathbf{q}}}^{c} + (\eta_v \underline{\mathbf{K}}^{ec} - i\Omega \underline{\mathbf{G}}^{ec})\dot{\vec{\mathbf{q}}}^{c} + [\eta_a \underline{\mathbf{K}}^{ec} - i(\Omega\eta_v + \eta_b)\underline{\mathbf{C}}^{ec}]\vec{\mathbf{q}}^{c} = 0 \tag{2.58}$$

2.2 Disk Nodal Equations

Since the beam element has been discretized into nodal degrees of freedom, so long as the locations of disks in the model are chosen to coincide with one of these nodal locations, the expressions for stiffness and inertia can be directly combined with the global matrices at that node. The mass element is considered as a body at a point with inertia, gyroscopic moments, and unbalance considered as external forces.

$$\underline{\mathbf{M}}^{d}\ddot{\vec{\mathbf{q}}}_k + \Omega \underline{\mathbf{G}}^{d}\dot{\vec{\mathbf{q}}}_k = \Omega^2 \vec{\mathbf{F}}^{d} \tag{2.59}$$

The superscript ^{d} in the above equation represents that the matrix or array is for a disk, and the subscript $_{k}$ on the displacement array indicates the array is only displacements for a single node, written out as: $\vec{\mathbf{q}}_k = [v, w, \psi, \theta, u, \phi]^{\intercal}$. The matrices and forcing array of (4.6) are as follows

$$\begin{cases} \underline{\mathbf{M}}^{d} = \begin{bmatrix} \rho Al & 0 & 0 & 0 & 0 & 0 \\ 0 & \rho Al & 0 & 0 & 0 & 0 \\ 0 & 0 & \rho I_z & 0 & 0 & 0 \\ 0 & 0 & 0 & \rho I_y & 0 & 0 \\ 0 & 0 & 0 & 0 & \rho Al & 0 \\ 0 & 0 & 0 & 0 & 0 & \rho J_x \end{bmatrix} \underline{\mathbf{G}} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \rho J_x & 0 & 0 \\ 0 & 0 & -\rho J_x & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \\ \vec{\mathbf{F}}^{d} = \begin{Bmatrix} \rho Al\epsilon \cos(\Omega t + \delta_\varepsilon) \\ \rho Al\epsilon \sin(\Omega t + \delta_\varepsilon) \\ -\rho(I_y - J_x)\chi \sin(\Omega t) \\ \rho(I_z - J_x)\chi \cos(\Omega t) \\ 0 \\ 0 \end{Bmatrix} \end{cases} \tag{2.60}$$

Disk unbalance is caused by an eccentricity, or a distance between the axis of rotation and the center of mass of the disk. Eccentricity is represented here as ϵ and is equivalent to that geometric distance. The unbalance moments, third and fourth equations of $\vec{\mathbf{F}}^d$ in (2.60), are caused by the skew angle, χ, which is the angle the disk major axis forms with the axis of rotation. A more detailed explanation of the moment function from skew angle can be found in [18]. The major axis is the axis normal from the disk face from which the polar moment of inertia, J_x is defined.

2.2.1 Disk in complex coordinates

Disk equations of motion (4.6) are converted to the complex plane in the same manner the beam element was derived in §2.1.7

$$\underline{\mathbf{M}}^{dc}\ddot{\vec{\mathbf{q}}}_k^c - i\Omega\underline{\mathbf{G}}^{dc}\dot{\vec{\mathbf{q}}}_k^c = \Omega^2\vec{\mathbf{F}}^{dc} \tag{2.61}$$

where,

$$\underline{\mathbf{M}}^{dc} = \begin{bmatrix} \rho Al & 0 \\ 0 & \rho I \end{bmatrix}, \quad \underline{\mathbf{G}}^{dc} = \begin{bmatrix} 0 & 0 \\ 0 & \rho J_x \end{bmatrix}, \quad \vec{\mathbf{F}}^{dc} = \begin{Bmatrix} \rho Al\epsilon e^{i\delta_\epsilon} e^{i\Omega t} \\ \rho(I - J_x)\chi e^{i\Omega t} \end{Bmatrix}$$

2.3 Bearing Nodal Equations

Bearings in this work are to be considered massless points of stiffness and damping acting at a node. Represented by the local equations of motion

$$\underline{\mathbf{D}}^b\dot{\vec{\mathbf{q}}}_k + \underline{\mathbf{K}}^b\vec{\mathbf{q}}_k = 0 \tag{2.62}$$

The superscript ^{b} indicates the matrix is for a bearing. A simple model for the stiffness and damping is used. Structural damping of the bearing is considered to be proportional to the stiffness, Raleigh Damping for the local bearing system. The

stiffness matrix is comprised of only transverse stiffness terms, as

$$\underline{\mathbf{K}}^b = \begin{bmatrix} k_{yy} & k_{yz} & 0 & 0 & 0 & 0 \\ k_{zy} & k_{zz} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad \underline{\mathbf{D}}^b = a\underline{\mathbf{K}}^b \tag{2.63}$$

The stiffness is typically simplified further to represent an orthotropic bearing, where $k_{yz} = k_{zy} = 0$ or further yet as a isotropic bearing, where $k_{yz} = k_{zy} = 0$ & $k_{yy} = k_{zz} = k$. Generally, these unknown parameters of stiffness are determined by changing the values to achieve the correct natural frequencies of the system.

2.3.1 Bearing in Complex Coordinates

Following the same methodology as sections §2.1.7 & §2.2.1, the equation of motion in the complex plane is

$$\underline{\mathbf{D}}^{bc}\dot{\vec{\mathbf{q}}}_k^c + \underline{\mathbf{K}}^{bc}\vec{\mathbf{q}}_k^c = 0 \tag{2.64}$$

where,

$$\underline{\mathbf{K}}^{bc} = \begin{bmatrix} k & 0 \\ 0 & k \end{bmatrix}, \quad \underline{\mathbf{D}}^{bc} = a\underline{\mathbf{K}}^{bc}$$

k is the isotropic bearing stiffness. It is possible to define the complex system with anisotropic terms of stiffness, but the complexity added to the system outweighs the benefit of using the complex plane in the first place.

2.4 Assembly of the Global System of Equations

The matrices in the global system of equations are determined using the direct approach of taking the summation of the inertia, damping, stiffness, or force at each degree of freedom.

2.4.1 Assembly In the Real Coordinate System

Combining all beam elements, disks, and bearings leads to the following global equations of motion in real coordinates.

$$\underline{\mathbf{M}}\ddot{\vec{\mathbf{q}}} + \underline{\mathbf{G}}\dot{\vec{\mathbf{q}}} + \underline{\mathbf{K}}\vec{\mathbf{q}} = \Omega^2\vec{\mathbf{F}} \tag{2.65}$$

where

$$\begin{cases} \underline{\mathbf{M}} &= \underline{\mathbf{M}}^e_G + \underline{\mathbf{M}}^b_G + \underline{\mathbf{M}}^d_G & \underline{\mathbf{D}} &= \eta_v\underline{\mathbf{K}}^e_G + \Omega\underline{\mathbf{G}}^e_G + \Omega\underline{\mathbf{G}}^d_G + \underline{\mathbf{D}}^b_G \\ \underline{\mathbf{K}} &= \eta_a\underline{\mathbf{K}}^e_G + (\Omega\eta_v + \eta_b)\underline{\mathbf{C}}^e_G + \underline{\mathbf{K}}^b_G & \vec{\mathbf{F}} &= \vec{\mathbf{F}}^d_G \end{cases}$$

Subscript $_G$ indicates the matrix is in the global coordinate system that contains all of the degrees of freedom. Care must be taken here to associate the correct degrees of freedom, and recognize that some of the matrices are elemental and others are nodal.

2.4.2 In the Complex Coordinate System

Performing the same summation as in the real coordinate system, but using complex coordinates results in the following global equations of motion. There are half as many equations in this expression than in the equivalent expression in real coordinates 2.65.

$$\underline{\mathbf{M}}\ddot{\vec{\mathbf{q}}}^c + \underline{\mathbf{D}}\dot{\vec{\mathbf{q}}}^c + \underline{\mathbf{K}}\vec{\mathbf{q}}^c = \Omega^2\vec{\mathbf{F}} \tag{2.66}$$

where,

$$\begin{cases} \underline{\mathbf{M}} &= \underline{\mathbf{M}}^{ec}_G + \underline{\mathbf{M}}^{dc}_G + \underline{\mathbf{M}}^{bc}_G & \underline{\mathbf{D}} &= \eta_v\underline{\mathbf{K}}^{ec}_G - i\Omega(\underline{\mathbf{G}}^{ec}_G + \underline{\mathbf{G}}^{dc}_G) + \underline{\mathbf{D}}^{bc}_G \\ \underline{\mathbf{K}} &= \eta_a\underline{\mathbf{K}}^{ec}_G - i(\Omega\eta_v + \eta_b)\underline{\mathbf{C}}^{ec}_G + \underline{\mathbf{K}}^{bc}_G & \vec{\mathbf{F}} &= \vec{\mathbf{F}}^{dc}_G \end{cases}$$

2.5 Analysis of the Resulting Model

A benefit of the finite element method is the resulting general linear Ordinary Differential Equations (ODEs) that are left to solve in equations (2.65) & (2.66). Many

techniques exist to provide frequency and time domain information on the solution of this system of equations. In their current state, equations (2.65) & (2.66) may be analyzed using frequency domain analysis techniques to be discussed in §3. On the other hand, in order to provide time domain solutions, numerical integration of these ODEs must be conducted. Details of the process of numerical integration is out of the scope of this work. Resulting time domain solutions of the ODEs for a specific nodal location can then be processed using the techniques outlined in §1. This time domain method of processing the FEM model is vital for analyzing non-linear differential equations because frequency domain analysis of non-linear differential equations often produces disjointed results that are difficult to extrapolate useful information from. Non-linear effects may result from a more detailed model of bearings, anisotropic beam elements, loose fittings, and many other sources.

Chapter 3

FREQUENCY DOMAIN ANALYSIS

Eigenvalues and eigenvectors of equations of motion will be used to provide information on the modal content of the system being analyzed. Eigenvalues can provide information on the natural frequencies, damping, and stability of the modes of vibration. Eigenvectors will be used to describe the shape of each of these modes of vibration including relative phase information for each nodal location. Frequency response functions will describe amplitude of response, phase lag of response from forcing input, and further stability information on modes. Inspiration for the content in this section came about as a synthesis of content provided on frequency domain analysis in [4], & [10].

Rotordynamic analyses are often interested in the dependence of the system on the spin speed Ω. The inclusion of this parameter as a time dependent variable would make the solutions of the system much more difficult. In this work, the spin speed is considered to be constant in each operation, taken in a series of operations as the spin speed is changed. In this way, the effect of spin acceleration is not taken into account, but the dependence on spin speed is approximated. For the vast majority of rotordynamic systems this approximation is sufficiently accurate, especially when considering that it is not a very common neccessity to ramp up quickly in speed.

For a simple critical speed computation, assuming the form of a homogeneous solution $\vec{\mathbf{q}} = e^{i\Omega t}$ while neglecting all damping and forcing in the equations of motion will provide the dynamic undamped forced whirling matrix. Exploring the eignevalues of this equation will give the undamped critical speeds of the system. In the case that the natural frequencies (whirl speeds) need to be calculated independent of spin speed, like in the Campbell diagram, a solution of the form $\vec{\mathbf{q}} = e^{st}$ is assumed so that the

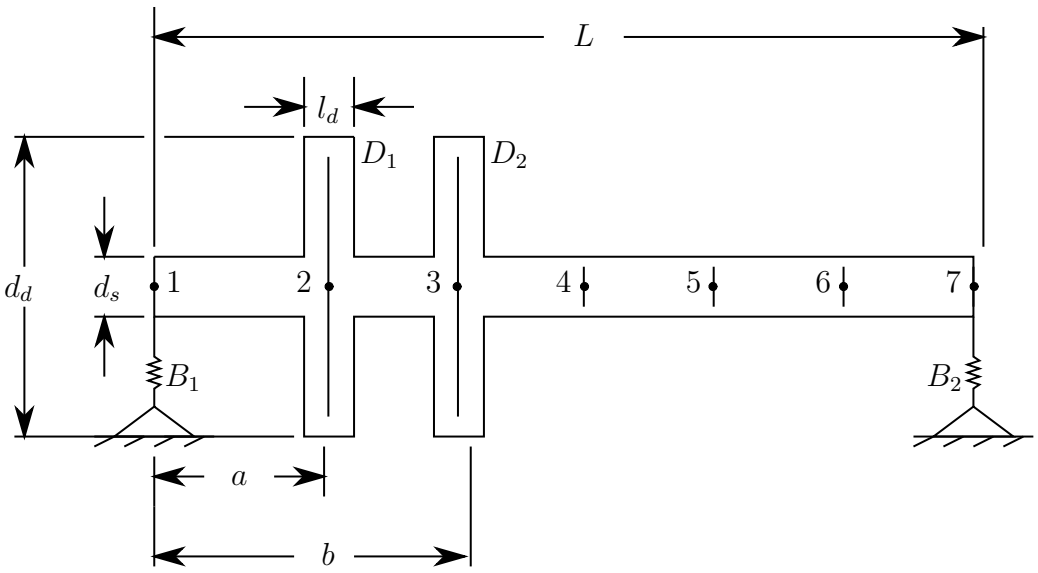


Figure 3.1: Diagram of Two disk model example problem.

spin speed and whirl speed can vary independently. Eigenvalues of this dynamic free whirling matrix will provide complex numbers of which the real part is proportional to damping, and the imaginary is damped natural frequencies of the system. Particular integrals of the form $\vec{\mathbf{q}} = e^{i\omega t}$ & $\vec{\mathbf{q}} = e^{i\Omega t}$ will be assumed to find the frequency response.

Details of the different analyses to follow will be accompanied by an example problem to demonstrate the results. The problem of interest is a simple two disk rotor system depicted in Figure 3.1. The geometry and material properties are listed in Table 3.1. The rotor is discretized as in figure 3.1 into 6 elements with geometry of $L = 1.8[m]$, $a = .3[m]$, & $b = .6[m]$ and bearings located at the ends of the shaft.

3.1 State Space Representation and the Eigenvalue Problem

The system can be represented in state space where,

$$\dot{\vec{\mathbf{z}}} = \underline{\mathbf{A}}\vec{\mathbf{z}} + \underline{\mathbf{B}}\vec{\mathbf{w}}, \quad \vec{\mathbf{z}} = \left\{ \begin{array}{c} \dot{\vec{\mathbf{q}}} \\ \vec{\mathbf{q}} \end{array} \right\} \tag{3.1}$$

Table 3.1: Properties of disks, shaft elements, and bearings of the example problem.

	$\rho[\frac{kg}{m^3}]$	$d_s[m]$	ν	$E[Pa]$	$\eta_v[s]$	η_h
Shaft	7850	0.1	0.3	210×10^9	0.0002	0
	$\rho[\frac{kg}{m^3}]$	$d_d[m]$	$l_d[m]$			
Disks; $\mathbf{D_1}$, $\mathbf{D_2}$	7850	0.6	0.1			
	$k_x[\frac{N}{m}]$	$k_y[\frac{N}{m}]$	$c_x[\frac{Ns}{m}]$	$c_y[\frac{Ns}{m}]$		
Bearings; $\mathbf{B_1}$, $\mathbf{B_2}$	1×10^7	1×10^7	100	100		

$\underline{\mathbf{B}}\vec{\mathbf{w}}$ is an input into the system, which here will represent the forces due to unbalance. Solving for $\underline{\mathbf{A}}$ & $\underline{\mathbf{B}}\vec{\mathbf{w}}$

$$\underline{\mathbf{A}} = \begin{bmatrix} -\underline{\mathbf{M}}^{-1}\underline{\mathbf{D}} & -\underline{\mathbf{M}}^{-1}\underline{\mathbf{K}} \\ \underline{\mathbf{I}} & \underline{\mathbf{0}} \end{bmatrix} \quad \& \quad \underline{\mathbf{B}}\vec{\mathbf{w}} = \Omega^2 \begin{Bmatrix} \underline{\mathbf{M}}^{-1}\vec{\mathbf{F}} \\ 0 \end{Bmatrix} \tag{3.2}$$

This equation is valid in both the complex and real coordinate plane. With the use of complex coordinates, the state vector is represented by $\vec{\mathbf{z}} = [\dot{\vec{\mathbf{q}}}^c, \vec{\mathbf{q}}^c]^\intercal$. $\underline{\mathbf{I}}$ & $\underline{\mathbf{0}}$ are appropriately sized to match the matrices in the system of choice. $\underline{\mathbf{A}}$ is the dynamic matrix of the system, which represents the dynamics in a single expression. Now in the form of a first order linear differential equation, numerical integration and many other numerical analysis techniques may be applied.

Using equation (3.1) and assuming a solution of $\vec{\mathbf{z}} = \vec{\mathbf{\Theta}}e^{st}$ while neglecting the forcing term $\underline{\mathbf{B}}\vec{\mathbf{w}}$ leads to the eigenvalue problem

$$(\underline{\mathbf{A}} - s)\vec{\mathbf{\Theta}} = \underline{\mathbf{0}} \tag{3.3}$$

$\vec{\mathbf{\Theta}}$ is the vector containing the eigenvectors. Since the state in which these equations are defined is the combination of displacement and velocity, the eigenvectors are defined as a set corresponding to both displacement and velocity. This effectively

duplicates the set as

$$\vec{\mathbf{\Theta}} = \left\{ \begin{array}{c} \vec{\theta}_{\dot{\vec{\mathbf{q}}}} \\ \vec{\theta}_{\vec{\mathbf{q}}} \end{array} \right\} = \left\{ \begin{array}{c} s\vec{\theta}_{\vec{\mathbf{q}}} \\ \vec{\theta}_{\vec{\mathbf{q}}} \end{array} \right\}$$

3.2 Dynamic Response

The dynamic response of the system to unbalance can be achieved by substituting the particular integral $\vec{\mathbf{q}} = \vec{\mathbf{q}}_0 e^{i\Omega t}$ into the equation of motion (2.65),(2.66).

$$\left\{ \begin{array}{ll} \vec{\mathbf{q}}_0 & = (-\Omega^2 \underline{\mathbf{M}} + i\Omega \underline{\mathbf{D}} + \underline{\mathbf{K}})^{-1} \Omega^2 \vec{\mathbf{F}} \\ \vec{\mathbf{q}}_0 & = H(\Omega) \end{array} \right. \tag{3.4}$$

where $H(\omega)$ is called the complex frequency response of the system due to unbalance. The frequency response acts as a transfer matrix that converts inputs, $\vec{\mathbf{F}}$, into outputs $\vec{\mathbf{q}}_0$. Use of this frequency response function with real coordinates requires a multiplication of the complex unit i to one of the orthogonal directions to ensure correct phase information. For the coordinate system set in §2.1 this results in i times w and i times ψ. This multiplication reflects the fact that w lags v by 90 degrees and ψ lags θ by 90 degrees in reference to a counterclockwise Ω. Values of H are complex with the absolute part representing the magnitude of the response, and the angle representing the phase delay of the response from the input. If the input phase is known in terms of shaft angle, then the phase delay difference of the response angle from the shaft angle of the input can be interpreted as a shaft angle output.

The response can also be calculated using the particular integral $\vec{\mathbf{q}} = \vec{\mathbf{q}}_0 e^{i\omega t}$ where ω is the independent whirl frequency. Then the response is found as the transfer response of the arbitrary oscillation, while spin speed remains independent.

For the example problem defined in §3 the unbalance response to a disk b unbalance of $2 \times 10^{-5}[m]$ is shown in Figure 3.2. Note that the phase lag angle and the amplitude of the response are plotted side by side forming the Bode diagram. For this

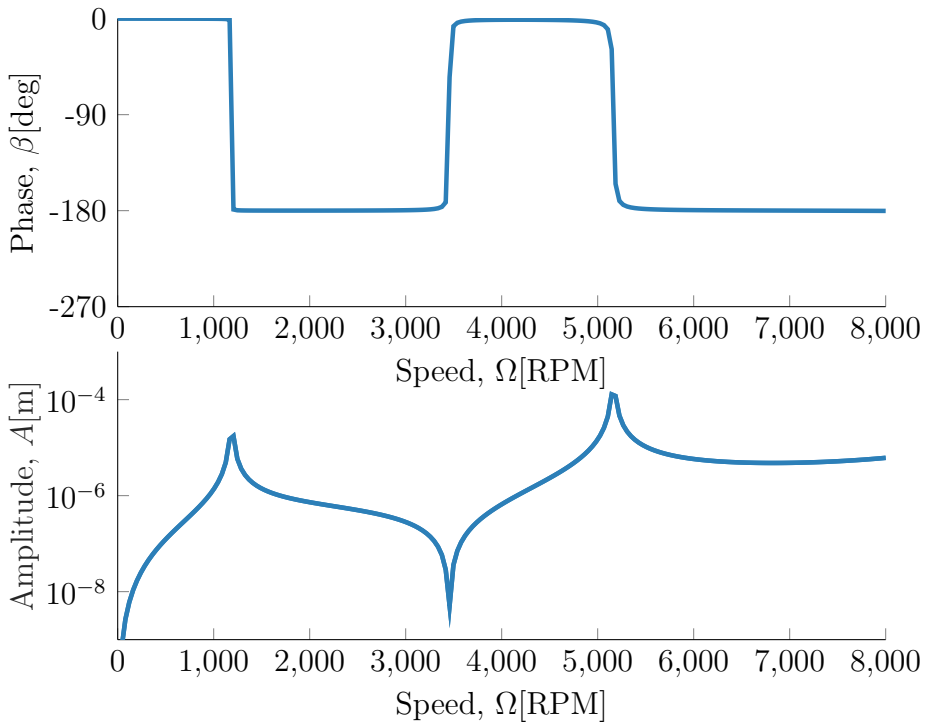
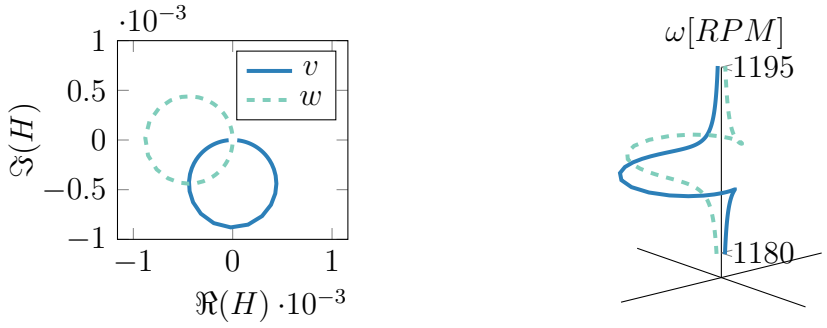
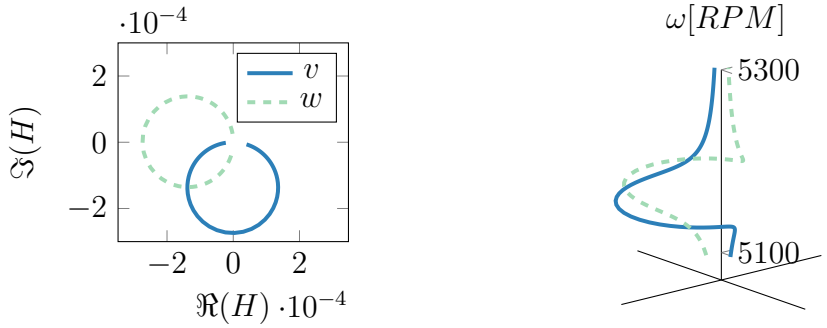


Figure 3.2: Bode diagram of the second disk subject to an unbalance at the second disk. Amplitudes of v & w are identical for this asymmetric system, phase of w would be lagging v by 90[deg].

same set of data, real and imaginary parts of the complex frequency response may be plotted on the complex plane to form the Nyquist diagram. Figure 3.3 contains a seperate diagram for each mode. During a natural frequency, if one of the circles traverses in the counterclockwise direction that mode is deemed unstable. Also, if the path of one circle crosses the path of the other pertaining to the orthogonal direction, the whirl is in the negative direction. Using the particular integral of $\vec{\mathbf{q}} = \vec{\mathbf{q}}_0 e^{i\omega t}$ instead of the synchronous solution, provides whirl frequencies independent of spin speed. This allows for the investigation of all modes of vibration at a specific spin speed. For instance, if the spin speed is set to $4000[RPM]$ and the Nyquist plot for the first mode is revisited (Figure 3.4) it is evident that the path traced is in the counter-clockwise direction indicating instability of mode 1 at the spin speed of $4000[RPM]$.



(a) Nyquist Diagram for the second mode in the speed range $1180 < \Omega < 1195[RPM]$.



(b) Nyquist Diagram for the second mode in the speed range $5100 < \Omega < 5300[RPM]$.

Figure 3.3: Nyquist plots for the first two modes at node 6 for the example two disk problem.

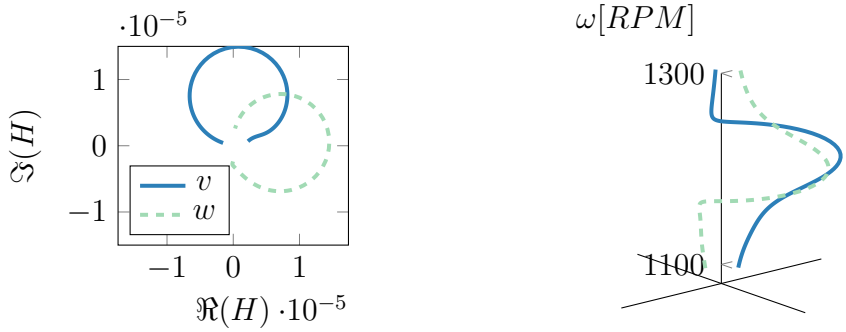


Figure 3.4: Nyquist Diagram for the first mode at node 6 in the whirl speed range $1100 < \omega < 1300$[RPM] at a spin speed, $\Omega = 4000$[RPM]. Counter-clockwise path indicates instability.

3.3 Roots Locus and Stability Analysis

One valuable facet of a rotordynamic model is its ability to predict instability. Unstable operation of rotor systems can lead to failures and unsafe operating conditions. Before the advent of modern predictive models for rotating machinery, it was not common to operate a machine above the first critical speed. Internal damping and other rotating damping can cause a subsynchronous whirl when operating above the first critical speed, this would often lead to failure of machines. The rotating damping effect is worse for machines with stiff bending modes, and is often instigated by loose fittings, shrink fits, and couplings. As mentioned in §2.1.6, the effect of external rotating damping sources will not be investigated here. Rather, the stability will be tested by modeling the internal damping of the rotor due to viscous heat production during loading and unloading of the beam in bending. See §2.1.6 for a more detailed explanation and derivation.

Roots Locus is the plot of eigenvalues on the $\Re - \Im$ plane as some value they are dependent on changes. In rotordynamic analysis the dependent variable is often the spin speed Ω. Using the definition of the state space eigenvalue problem set out in (3.3), the eigenvalues s represent a complex set of which the real part is proportional to damping and the imaginary is the damped natural frequency. The roots locus is useful in determining stability of the system, and which mode is responsible for the instability. Using the eigenvalues s of the eigenvalue problem defined in equation (3.3) on the example problem defined in §3 a plot of the roots locus can be found as in figure 3.5. Internal damping coefficient, η_v, is added to the system at the value of $0.0002[s]$. The first three modes are presented with the negative complex eigenvalue being represented in the positive imaginary axis alongside each positive mode of vibration. In completely symmetric systems, positive and negative modes have identical eigenvalues at a spin speed of zero. As the spin speed increases, positive

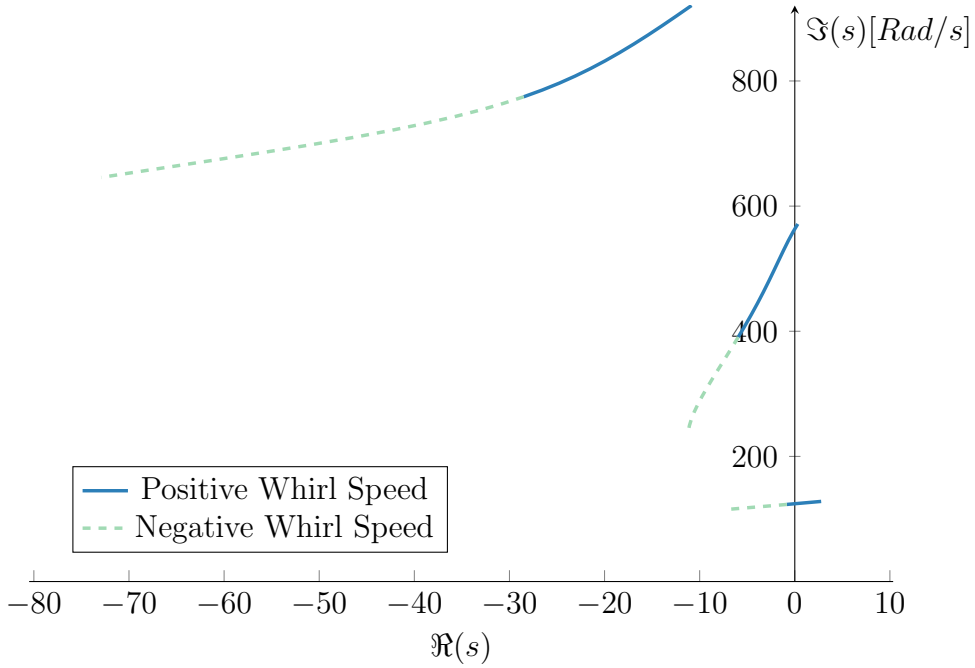


Figure 3.5: Roots Locus of the example problem with an internal damping coefficient of $0.0002[s]$**.**

modes move in the positive real direction–becoming more unstable. On the other hand, the negative frequencies move in the negative real direction–becoming more stable. This phenomena of the positive modes becoming more unstable and the negative becoming more stable is a general trend, but not a rule, rotating damping defined in §2.1.6 assists in this trend. It is evident by looking at figure 3.5 that the system becomes unstable at some point, as many of the eigenvalues are in the positive real region of the plane. According to the plot, the first and second modes both become unstable as speed increases. Gyroscopic effects tend to increase the whirl frequency (imaginary part of eigenvalue) of the positive mode, while the opposite effect is seen in the negative mode. In general, the effect of non-rotating damping is to rotate modes in the counter-clockwise direction, at first away from the imaginary axis, and eventually on to the real axis when the mode becomes over-damped. The goal of the application of the magnetic bearing in §4 will be to provide enough non-rotating damping to render as many modes as possible over-damped.

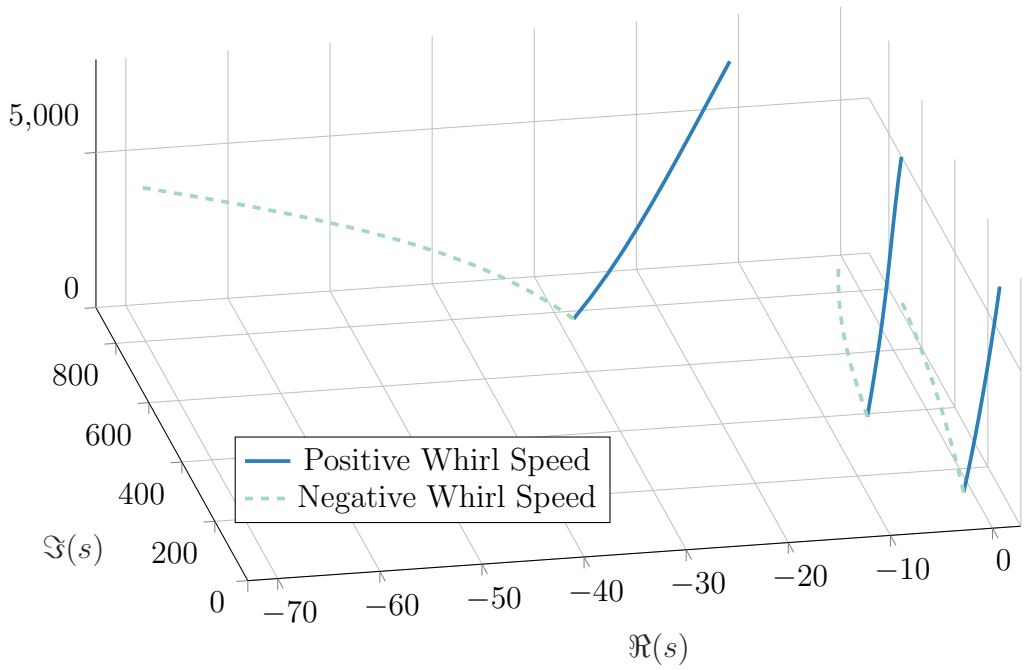


Figure 3.6: Roots Locus of the example problem in 3-D with an internal damping coefficient of $0.0002[s]$**.**

The Roots Locus for the example problem is also presented in three dimensions to lend in the understanding of the relationship with spin speed. This can be seen in Figure 3.6. A direct method of measuring the stability of the system is to observe only the real part of the eigenvalues of the dynamic matrix as they vary with spin speed. Since the solution was assumed the be of the form $\vec{\mathbf{z}} = \vec{\mathbf{\Theta}} e^{st}$, then the eigenvalues represent the values of the complex exponent s. Splitting s into its real and complex components as $s = \sigma + i\omega_d$, and plugging into $\vec{\mathbf{z}}$ gives: $\vec{\mathbf{z}} = \vec{\mathbf{\Theta}} e^{\sigma + i\omega_d}$. By inspection, it is evident that when the real part is positive, the response will grow without bound–an unstable system.

Damping is commonly represented as a the ratio of the real part of the eigenvalue to the natural frequency as

$$\zeta = \frac{-\sigma}{\omega_n} \ , \ \omega_n = \sqrt{\sigma^2 + \omega_d^2} \tag{3.5}$$

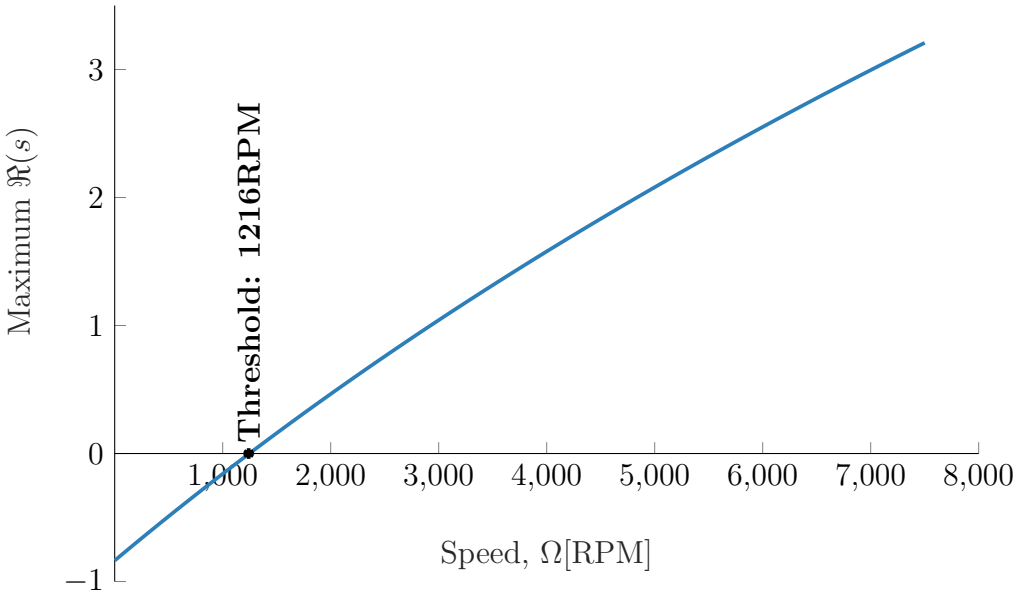


Figure 3.7: Stability plot of the example problem.

where ζ is the damping ratio, and ω_n is the natural frequency. The stability margin, or the range of speeds through which the system remains stable can be determined by plotting the maximum real part, $\Re(s)$, of the set against spin speed. The point at which $\Re(s)$, or σ, crosses the real axis is the threshold of stability. The plot which represents this is termed the Stability Margin plot, and is shown in 3.7 for the example problem defined in §3.

Finally, the stability can be analyzed using the damping coefficients(ζ) plotted against speed. This figure, shown in 3.8, indicates instability as when ζ drops below zero.

3.4 Campbell

The Campbell diagram correlates the spin speed and the whirl speed. Whirl speed is calculated as the imaginary part of the complex eigenvalue defined by the eigenvalue problem in equation 3.3. Spin speed is varied while calculating all of the critical whirl

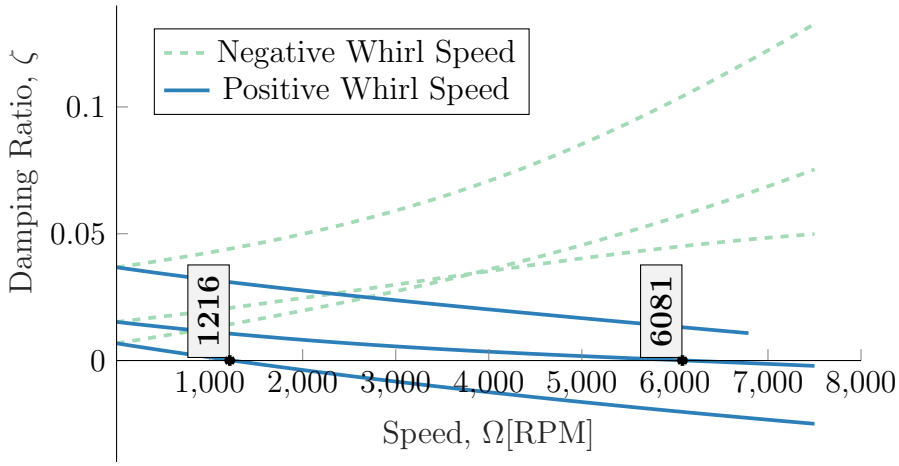


Figure 3.8: Damping ratio of the first three modes of the example problem, with indication of threshold of stability for each mode.

speeds at each step of spin speed. The campbell diagram for the example problem is given in figure 3.9. This diagram can be used to discern the critical speeds of the system by inspecting where the $\omega = \Omega$ synchronous line passes through positive and negative modes in this speed range. Here the synchronous line passes through three separate modes, each with a positive and negative whirl speed. Note: with the large influence of the gyroscopic effect in the example problem presented here, the positive and negative whirl speeds for the second and third modes diverge from one another rather quickly, but this is not the case for the first mode. A physical interpretation of this phenomena will be presented in §3.5

3.5 Shapes

Deformed shape of each mode can be predicted using the eigenvectors of the eigenvalue problem from equation 3.3. Eigenvectors contain displacement arrangement that corresponds to a natural frequency; this includes phase and relative amplitude of the generalized displacements from one another. A simple model will be utilized to conceptualize this idea, consider a rigid rod that can only translate in one direction at each end. The eigenvectors will contain all linearly independent combinations of the

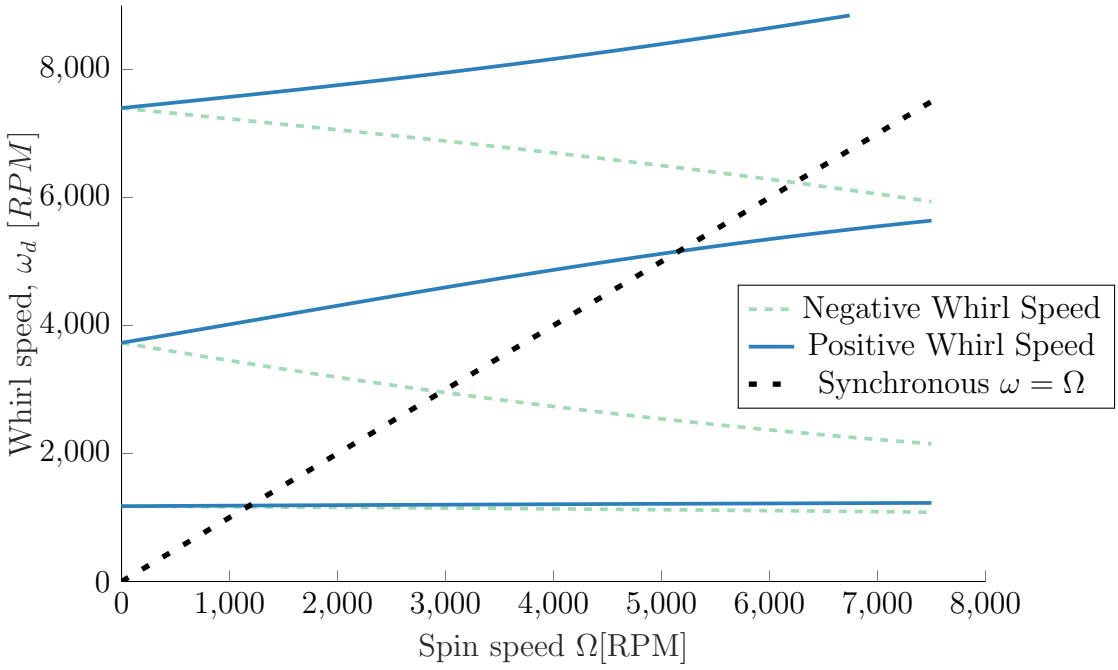


Figure 3.9: Campbell Diagram of the example problem.

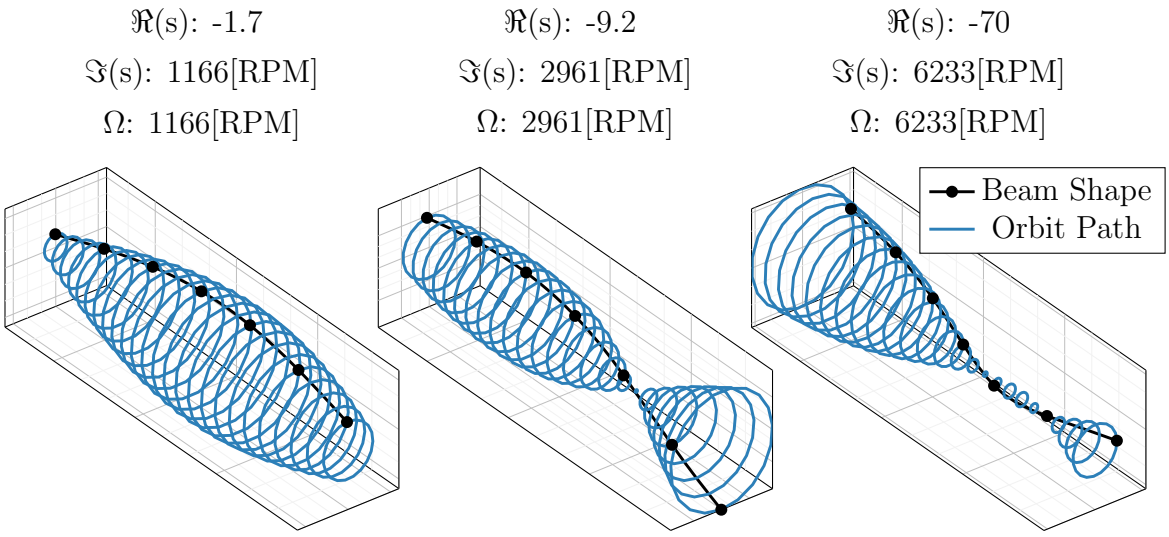
movement of the left side relative to the right. This would be in general; opposite left and right (i.e. one up, one down), and same left and right(i.e. both up, or both down). The bending modes of the beam operate in a similar manner but scaled up, and just like in the simple case, including the same number of modes as there are degrees of freedom in the system. Since the state space representation used for eigenanalysis contains $\dot{\vec{\mathbf{q}}}$, & $\vec{\mathbf{q}}$ the eigenvector will contain $2N$ modes. For a general ith mode of vibration, the eigenvector $\vec{\mathbf{\Theta}}_i$ contains the displacement information. But, since the portion of $\vec{\mathbf{\Theta}}_i$ pertaining to velocities, $\vec{\theta}_{\dot{\vec{\mathbf{q}}}i}$, is a linear combination of $\vec{\theta}_{\vec{\mathbf{q}}i}$, all of the information is contained in one of these arrays. Therefore, the interpolation of displacements for the ith mode is given by

$$\vec{\mathbf{u}}_i = \underline{\mathbf{N}}\vec{\theta}_{\vec{\mathbf{q}}i} \tag{3.6}$$

where, the value of $\vec{\mathbf{u}}_i$ is now a function of x. So, through the choice of an array of positions that span the length of the beam element the displacement, and phase angle distribution of the ith mode of vibration is determined. Amplitude in eah direction is

taken as the absolute value of the corresponding eigenvector value for that node and that degree of freedom. Phase angle is obtained from the angle of that same complex value. Phase angle and amplitude of an transverse pair may be used to create an orbit of a beam axis location x. Then all orbits along the beam axis can be stacked and a shape is formed as in figure 3.10.

Shape figures for the first 3 modes of the example problem are presented in Figures 3.10a,3.10b, and 3.10c. Note that the real and imaginary parts of the eigenvalues are listed along with the shapes. Also, note that the speed is chosen to coincide with the damped natural frequency so that this is the critical speed bending shape. Shapes of beam modes may be determined for any spin speed Ω.



(a) Mode Shape 1. (b) Mode Shape 2. (c) Mode Shape 3.

Figure 3.10: Modal shapes of the example problem.

A physical interpretation of the selective effect gyroscopic moments have on the whirl speeds of different mode shapes can be presented by taking the mode shapes into account. Recall that the disks in the example are mounted at nodal locations 2 & 3. In figure 3.10 it is evident that in the first mode, Figure 3.10a, both disks are translating together. As a matter of fact, most of all the points of the system are translating in phase. This type of mode is called a cylindrical mode for the

fact that as the shaft's path is traced, as it is in 3.10a, it forms a cylinder. The consequence of this type of motion is small transverse rotation angles, ψ & θ, of the mass elements. Since the gyroscopic moments are proportional to these angles, their magnitude is very low compared to transverse effects and as a result the whirl speeds are nearly unaffected by spin speed changes. On the contrary, modes 2 & 3 of figures 3.10b & 3.10c respectively, the angle of the disks and other mass elements is changing significantly through rotation. Because of this, large gyroscopic moments induce an out of phase force that stiffens positive whirl and softens negative whirl tendencies. Gyroscopic moments are larger in these types of modes where there is a point of inflection on the beam, commonly called an antinode of vibration. This mode is called a conical mode because the shape of the path forms one or many cones.

Chapter 4

SYNTHESIS IN EXAMPLE OF A MAGNETIC BEARING ON AN OVERHUNG ROTOR

Analysis and modeling techniques of the previous chapters will now be put to use in a practical example. The goal of which will be to reduce vibration on an overhung disk rotor system with the use of an Active Magnetic Bearing(AMB). An experimental test rig, not unlike the system used in the experimental example of §1.2, will be used to calibrate a finite element theoretical model. Then, the theoretical model will be extended to include an AMB near the overhung disk. The model will be evaluated for stability, and parameters of the control algorithm for the AMB will be varied to attempt to eliminate modes and stabilize the system.

4.1 Physical System Description

The rotor system of interest is depicted in Figure 4.1. Geometric parameters are listed in Table 4.1. The springs at nodes 1 & 4 are intended to represent bushings, portion at node 6 is the rotating disk, and nodal numbers are indications of how the rotor will be discretized for the finite element model.

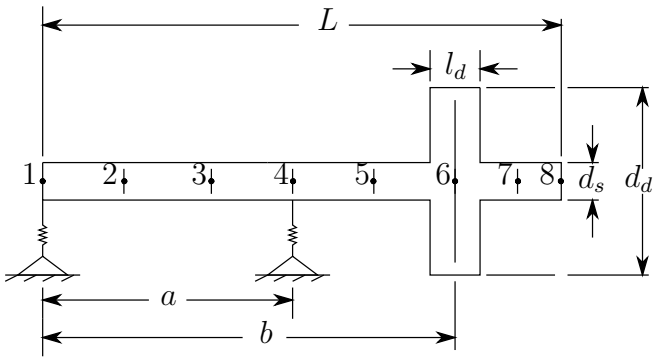


Figure 4.1: Overhung rotor system diagram.

Table 4.1: Geometric parameters of the overhung rotor system.

$L[m]$	$a[m]$	$b[m]$	$l_d[m]$	$d_d[m]$	$d_s[m]$
0.5	0.23	0.13	0.025	0.075	0.01

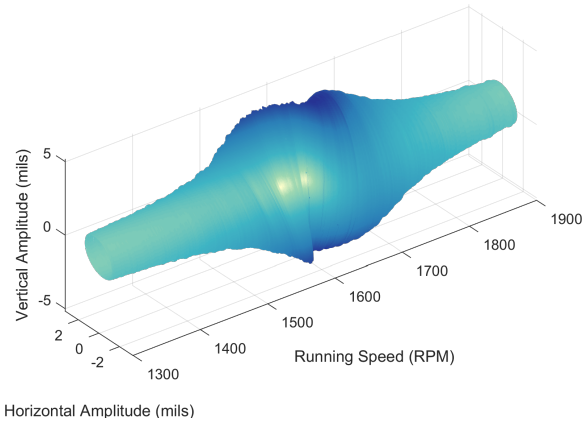


Figure 4.2: 3D Orbit of the experimental overhung rotor system.

4.2 Experimental Results

The rotor system was tested in a start-up from slow roll to 3000$[RPM]$. Data shown here is taken from 1000$[RPM]$ to 2000$[RPM]$. A set of orthogonal eddy current position sensors placed near the disk on the outboard side were used to measure the position of the shaft throughout the start-up. Position data was recorded at a sampling rate of 128000$[Hz]$ with no processing applied, this sampling rate is much higher than the required rate to avoid aliasing. The resulting 3D orbit of the start-up is shown in Figure 4.2. Of note is the necking in the 3D orbit during the natural frequency that is indicative of high anisotropy inducing a negative whirl during the first natural frequency. Also resulting from the experiment is the full spectrum cascade plot of Figure 4.3, in which it is evident that synchronous vibration dominates the spectra. Though, for the production of the bode diagram (Figure 4.4) there was a benefit in clarity from filtering the data to synchronous speed.

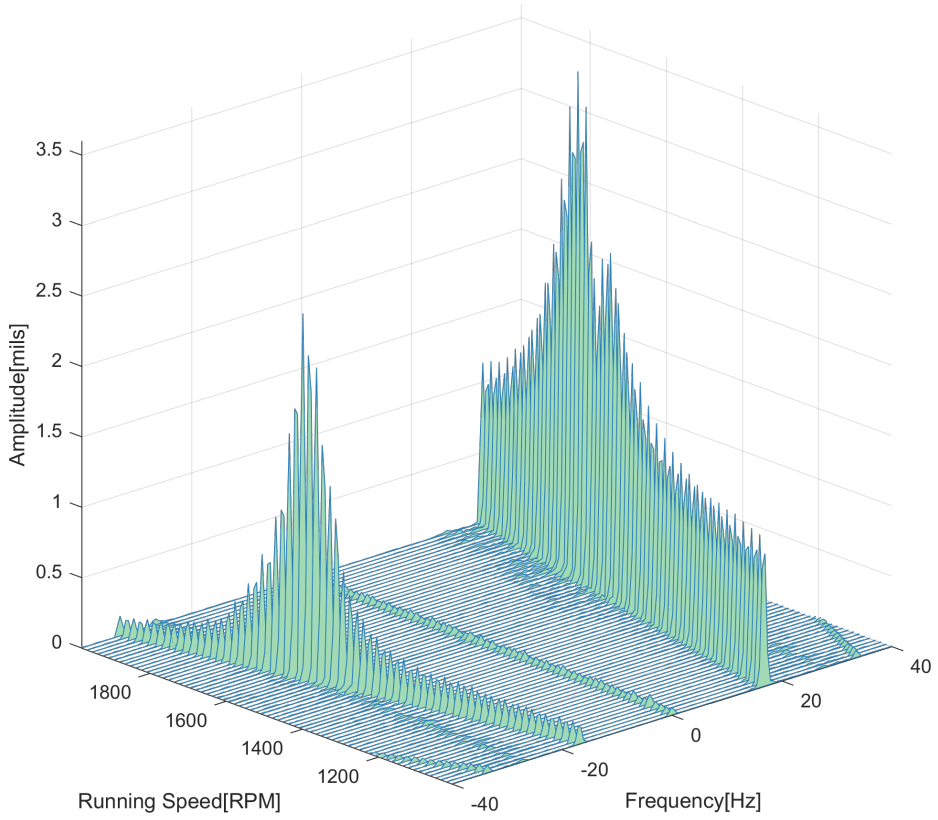


Figure 4.3: Cascade of the experimental overhung rotor system.

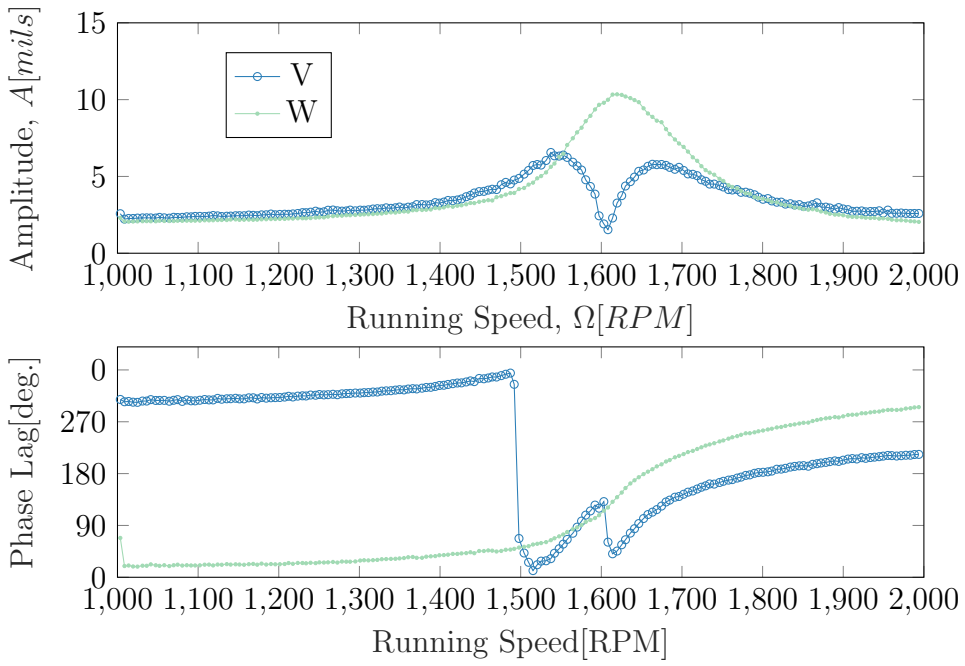


Figure 4.4: Bode diagram of the experimental overhung rotor filtered to 1X.

4.3 Theoretical Model

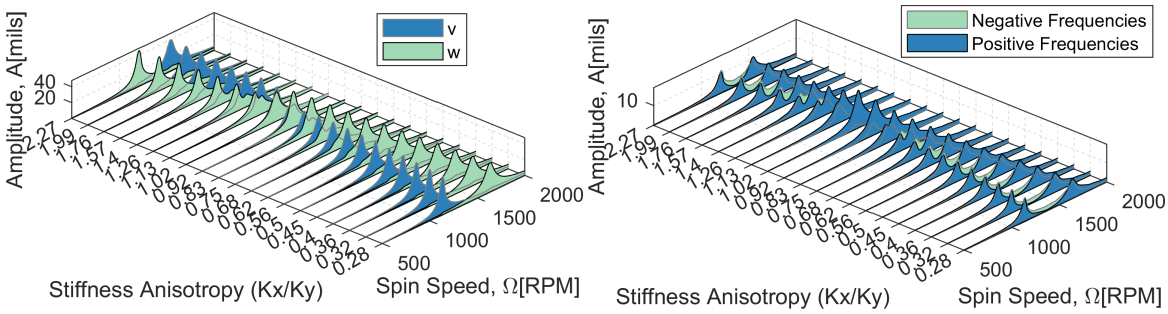
To create a finite element model for this rotor system the shaft will be discretized into 7 elements a disk at node 6 and bearings at nodes 1 and 4, as depicted in Figure 4.1. In the experiment, a small length of shaft continued after the overhung disk and has been included in this model. The AMB will be included as a nodal point element with with stiffness and damping to be derived in §4.3.1. Parameter values for the finite element model are provided in Table 4.2. Discovered values such as η_v and the stiffnesses of bearing are listed here, but the process for their determination is discussed. First the model is formed to match the experimental results. Known parameters, such as beam lengths, beam diameters, density of the material, and geometry of the disk and rotor are used to begin construction of the model. Then, guesses are made for the stiffnesses in the rotor bearings. The first natural frequency is calculated with the resulting model and its value compared to the experimentally

Table 4.2: Properties of disks, shaft elements, and bearings of the theoretical model.

	$\rho\left[\frac{kg}{m^3}\right]$	$r[m]$	ν	$E[Pa]$	$\eta_v[s]$	η_h
Shaft	7850	0.005	0.3	210×10^9	0.0002	0
	$\rho\left[\frac{kg}{m^3}\right]$	$r[m]$	$l[m]$			
Disks	7850	0.0375	0.025			
	$k_y\left[\frac{N}{m}\right]$	$k_z\left[\frac{N}{m}\right]$	$c_y\left[\frac{Ns}{m}\right]$	$c_z\left[\frac{Ns}{m}\right]$		
Bearing A	1.7×10^5	2.2×10^5	68	88		
Bearing B	2.04×10^5	2.64×10^5	81.6	105.6		

found natural frequency from the bode diagram (fig.4.4). Stiffness are then adjusted to better match the natural frequency, and this is repeated until the natural frequency of the model matches the experiment. After this process, the stiffness was determined to be around $2 \times 10^5\ \left[\frac{N}{m}\right]$.

It is evident by inspection of the Bode diagram for the experimental system (fig.4.4), and the 3D orbit, that there is anisotropy in the system, leading to the dip in amplitude of one plane of vibration. It is also known from inspection of the frequency spectrum in the cascade of figure 4.3 that in this speed range the orbit is in the opposite direction of the rotation–a phenomena only possible with anisotropy of the stiffness. Figures 4.5a & 4.5b demonstrate this affect anisotropy has on both the real coordinates, as well as positive and negative whirl amplitudes. In the real coordinates, cross coupling of from gyroscopic moments causes an interaction of the two peaks in amplitude. As the stiffness anisotropy approaches a value of 1, the vibration of the more flexible plane will split into two peaks. One peak coinciding with that planes natural frequency and one plane coinciding with the orthogonal natural frequency. During this change in anisotropy of stiffness, the peaks of both positive and negative amplitude spectrums split to coincide with either plane's natural frequency. When the two orthogonal planes have widely separated natural frequencies,



(a) Comparison of v & w as stiffness anisotropy is changed.

(b) Comparison of positive and negative orbit amplitudes as stiffness anisotropy is changed.

the negative amplitude spectrum is greater than the positive amplitude spectrum for the range of speed between the two natural frequencies.

Using the bode diagram, the stiffness anisotropy is adjusted until the shapes of the amplitudes and phases match the experimental results of figure 1.4. Then damping is added to the system to appropriately match the experimental results. The resulting bode diagram is Figure 4.6. Stiffnesses were determined to be $k_y = 1.7 \times 10^5[N/m]$ & $k_z = 2.2 \times 10^5[N/m]$. And now the resulting system is further described with an expanded speed range of $20000[RPM]$. This resulting model is described with the new Campbell diagram of Figure 4.7, the roots locus of Figure 4.8, and the first three mode shapes (Figures 4.9a, 4.9b, & 4.9c). Note that the first mode is conical in shape, but the disk is far from an antinode at node 6 so it does not experience significant gyroscopic moments. This idea is supported by the campbell diagram, fig. 4.7, as the first mode natural frequency does not change significantly over the speed range. On the other hand, the second mode has its antinode atode 6 so the disk experience maximum gyroscopic moments and the second mode critical speed is much more dependent on speed.

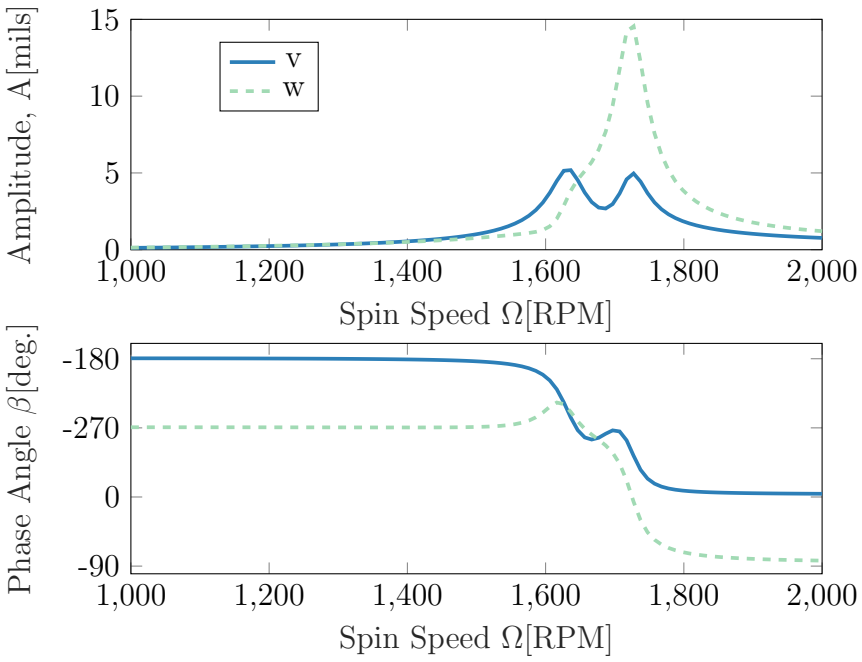


Figure 4.6: Bode Diagram of overhung system without AMB.

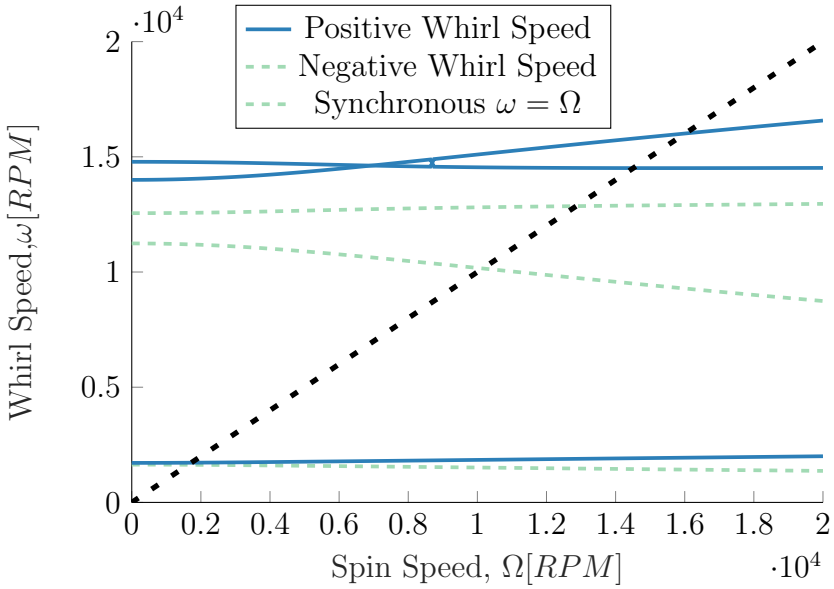


Figure 4.7: Campbell Diagram of the overhung system without AMB.

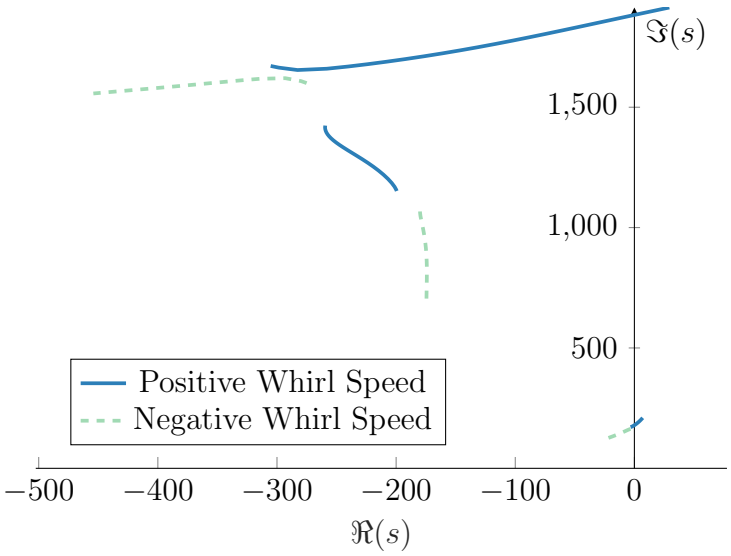


Figure 4.8: Roots Locus of overhung system without AMB.

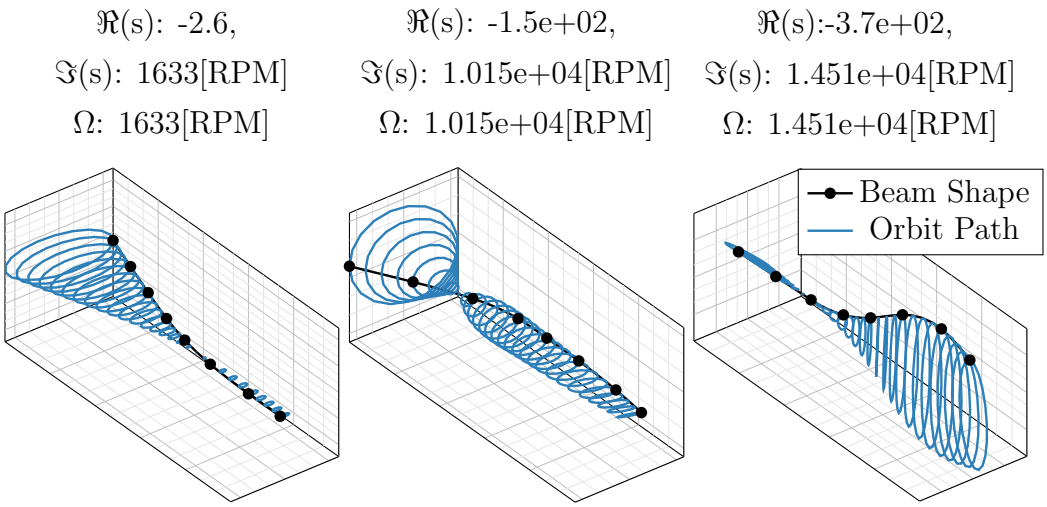
4.3.1 Active Magnetic Bearing

Magnetic pole-rotor relationship will be derived based on the detailed derivation in [5], with influence from [3]. Assumptions made in this model are as follows:

- Air gap between the magnet pole face and the rotor is vanishingly small compared to the diameter of the shaft.
- Flux leakage from the magnetic pole is negligible.
- Curvature of rotor surface under the pole face is ignored.
- A linear relationship of flux density and magnetic field is assumed.
- Hysteresis of the magnetic field is negligible.

These assumptions lead to a magnetic force due to coil current in the relationship of

$$F_m = \frac{-k_m i^2}{l_g^2} \tag{4.1}$$



(a) Mode Shape 1. (b) Mode Shape 2. (c) Mode Shape 3.

where, $k_m = \frac{\mu_0 A_p N^2}{4}$, and $\mu_0 = 4\pi \times 10^{-7}$ is the absolute permeability in free air, N is the number of coil turns, A_p is the pole face area, i is the supplied electrical current, and l_g is the air gap.

Consider a set of magnetic pole pairs in each the y & z directions (i.e. One on the left and one on the right; one on top and one on bottom), where each pole pair has its own electrical circuit. There are four total magnets in this model, two in each direction, and eight total poles where two form a magnet. All poles in the neutral position of the system will have a nominal air gap of g_0 and will be supplied by a bias current of i_0. Deviations of the current from this bias will be considered as i_y and i_z. Deviations of the position from this neutral position are the displacements of the rotor at this beam axis location, v, & w. Therefore, total current will be $(i_0 \pm i_y)$ & $(i_0 \pm i_z)$ for opposing poles in the y & z directions respectively. Similarly, total gaps are given by $(g_0 \pm v)$ & $(g_0 \pm w)$. Using these new definitions for the gap and current, and summing forces in the y & z directions leads to the total forces

$$F_y = K_m \left\{ \left(\frac{i_0 + i_y}{g_0 + v} \right)^2 - \left(\frac{i_0 - i_y}{g_0 - v} \right)^2 \right\} \;\&\; F_z = K_m \left\{ \left(\frac{i_0 + i_z}{g_0 + w} \right)^2 - \left(\frac{i_0 - i_z}{g_0 - w} \right)^2 \right\} \tag{4.2}$$

where, $K_m = k_m \cos(\alpha)$, and α is half of the angle between the poles of a magnet. Linearizing the magnetic force equations (4.2), while assuming the operating point for the system is where all positions and control currents are zero, results in

$$F_y = k_i i_y + k_y v, \ \& \ F_z = k_i i_z + k_z w \tag{4.3}$$

where, $k_i = 4K_m \frac{i_0}{g_0^2}$ is the current stiffness developed by the bias current, and $k_y = k_z = k_s = -4K_m \frac{i_0^2}{g_0^3}$.

4.3.1.1 Proportional Derivative Control

A control algorithm is used to control the current sent to each pair of poles based on the position (proportional) and velocity (derivative) of the rotor. It is assumed that each set of pole pairs will receive opposite currents to act as a unit. Current control is given by

$$i_y = -k_g(k_p v + k_v \dot{v}), \ \& \ i_z = -k_g(k_p w + k_v \dot{w}) \tag{4.4}$$

where, k_g is the power amplifier gain, k_p is the proportional gain, and k_v is the derivative gain. The total linearized force becomes

$$F_y = -(k_g k_i k_p - k_s)v - k_g k_i k_v \dot{v}, \ \& \ F_z = -(k_g k_i k_p - k_s)w - k_g k_i k_v \dot{w} \tag{4.5}$$

so then the stiffness of the AMB is $k_{mag} = (k_g k_i k_p - k_s)$ and the damping is $d_{mag} = k_g k_i k_v$. As an equation of nodal stiffness and damping in the finite element system it can be represented by

$$\underline{\mathbf{D}}^m \dot{\vec{\mathbf{q}}}_k + \underline{\mathbf{K}}^m \vec{\mathbf{q}}_k = 0 \tag{4.6}$$

where,

$$\underline{\mathbf{K}}^m = \begin{bmatrix} k_{mag} & 0 & 0 & 0 & 0 & 0 \\ 0 & k_{mag} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad \underline{\mathbf{D}}^m = \begin{bmatrix} d_{mag} & 0 & 0 & 0 & 0 & 0 \\ 0 & d_{mag} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

4.4 Addition of Magnetic Bearing to the Rotor Model

In order to measure the effectiveness of the AMB, the Stability of the theoretical model before the addition is shown in Figure 4.10a. The magnetic bearing added is modeled after a magnetic bearing that is currently in the lab at California Polytechnic State University. This theoretical exercise is intended to be followed by experimental verification not included in this work. The parameters used are listed in 4.3 as well as the control values whose determinations will be evaluated.

First the axial position of the AMB must be determined. By inspection of the mode shapes, it is evident that in the first mode (the mode we are most concerned about suppressing) a shape exists with increasing amplitude toward the end of the beam after the second bearing. It is also known that the source of vibration, the rotating disk, is located at nodal index six. With both these pieces of information, node seven is chosen as a starting point for the AMB for two reasons: having the AMB closer to the source of vibration reduces the phase lag between the source and the bearing, increasing the effectiveness of control; amplitude of vibration, according to the mode shape, is higher on the outboard side of the disk and placing the AMB on this side will minimize more vibration. Additionally, this location should significantly suppress the second mode of vibration as node seven is near the maximum amplitude of that mode as well.

In order to choose a value for bias current, i_b, and provide upper range estimates of the controller outputs, k_v, & k_p, the maximum expected current output of the control loop must be determined. i_b will be set so that under peak power output, in the expected speed range, the control current will push the total current output beyond the system limits.

Knowing that the amplitude of vibration, A, in the experimental system peaks at about 10 mils, or $2.54 \times 10^{-4}[m]$, the greatest possible velocity for synchronous vibration is determined using the simple equation $vel = (A/2)\omega$, where ω is the whirl speed in rad/s. Under the assumption of synchronous vibration, ω during the natural frequency is then $167.6[Rad/s]$. Leading to a velocity, vel, of $0.02[m/s]$. Also, another possible peak velocity occurs in the upper speed range. Knowing that the amplitude of vibration after the first natural frequency is equal to the eccentricity of the unbalance, with an eccentricity of 1×10^{-5}, and an upper speed of $15000[RPM]$, the velocity is calulated to be $0.008[m/s]$. Since the estimated velocity during the natural frequency is higher, it will be used to limit the output of the AMB controller. Total voltage supply of most digital to analog converters is limited to $10[V]$. So the control voltage is calculated to be within this range for the given velocity and positions that will be seen under synchronous vibration. This results in a limitation of the term k_v, which is proportional to the voltage control signal that is sent to the amplifier for conversion to control current. A value of $480[Vs/m]$ would max the converter, so the value to be determined must be less than this. With a similar, but separate, evaluation of the displacement leads to a maximum allowable value of $500000[V/m]$ for k_p–a value not anticipated to be necessary. The maximum force due to unbalance is $\epsilon m_d \Omega^2$, where ϵ is the eccentricity, and m_d is the mass of the disk. At the maximum speed expected of $15000[RPM]$, the force will be $24[N]$. To counteract this force with a single coil would require $1.25[A]$, or $0.625[A]$ per coil in a opposing pair. The amplifier gain is assumed to be programmable to $k_g = 1[A/V]$, this leads to a reasonable choice

of bias current at $0.5[A]$ to maintain a good resolution on the voltage output of the controller. Under the operating conditions described, the control voltage should not exceed $1.25[V]$, and will rest at an output of $0.5[V]$. Now in the next section, control parameters k_v & k_p will be determined to maximize stability of the system while also minimizing vibration.

To determine best derivative control constant (k_v), the proportional control constant (k_p) was set to zero. k_v is increased until the first mode on the Roots locus(fig. 4.8) moves away from the imaginary axis, becoming more damped. The resulting movement of the Roots on the Roots Locus is given in Figure 4.11. Note that because of bias current flowing through the bearing, a baseline stiffness of k_S is present in the system even withou k_p. None of the modes that appear on the roots locus of the system with $k_p = 0,\ \&\ k_v = 10[\frac{Vm}{s}]$ control algorithm cross the real plane– this demonstrates the stability of the new system. Stability of the system with the AMB addition is confirmed in the stability plot of Figure 4.10b. In fact from the roots locus with $k_v = 1$ to the roots locus with $k_v = 10$ the first mode of vibration is completely relegated to the imaginary axis and not reaching break-away for this entire speed range–this indicates that the first mode has been over-damped. With additional damping through the increase of k_v it would be possible to over-damp the second mode as well, but in the speed range given, it would not improve the performance.

Proportional control, k_p, was added after the ideal derivative control was determined, but it did not improve the performance of the controller. It is the case that the stiffness of the bearing inherent to the bias current is sufficient. In any case, proportional control in this scenario is only adding stiffness to the system, and with the objective being to minimize vibration, k_p is not effective. So the optimal control is determined to be with $k_p = 0,\ \&\ k_v = 10[Vs/m]$. The remaining parameters for this resulting controller are listed in table 4.3. It is worthwhile to note that this

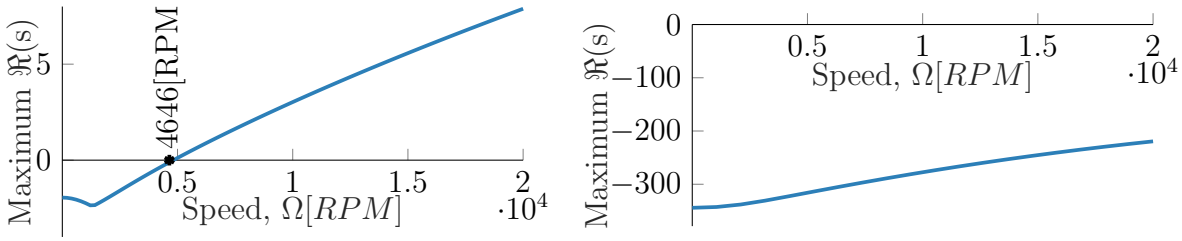
Table 4.3: Active Magnetic Bearing Parameters.

$\alpha[rad]$	$g_0[m]$	$i_0[A]$	$k_p[\frac{V}{m}]$	$k_v[\frac{Vs}{m}]$	$k_g[\frac{A}{V}]$	$N[\#]$	$A_p[m^2]$
$\frac{\pi}{8}$	2.5×10^{-3}	0.5	0	10	1	800	$\frac{5}{100*100}$

optimal control is standing on the basis that the feedback is of synchronous vibration only. In a scenario where there is significant sub or super-synchronous vibrations, this controller may exceed its voltage limit. It is recommended that the feedback signal be filtered to match rotor speed to ensure this scenario does not take place. Furthermore, without at least low-pass filtering of the feedback signal the control would certainly provide out of range signals due to the volatile nature of derivatives of discrete signals.

The frequency spectrum is provided showing the result of the AMB application in the bode diagram of Figure 4.12. Certainly it can be concluded that the AMB is successfully performing the desired task of reducing the vibration while also inproving the stability of the system.

The result from this synthesis exercise can be implemented on the actual experimental test rig with the AMB set to the control parameters suggested. The power of the finite element method in this application is the ability to move components around with ease. For instance, with the changing of just two parmeters in the input file for this model, a new simulation is created for complete levitation of the overhung rotor. The AMB is put in place of bearing b and the resulting frequency spectrum is plotted in Figure 4.13



(a) Stability plot of rotor without AMB, threshold of stability: 4646[RPM].

(b) Stability plot of rotor with AMB, indicates complete stability through speed range.

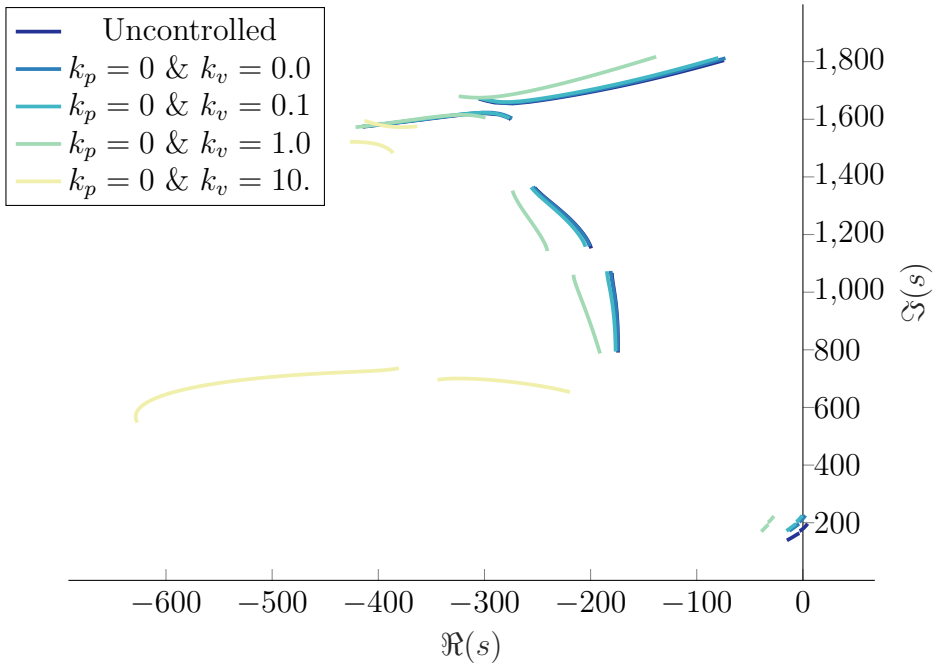


Figure 4.11: Roots locus of Overhung rotor system with varying k_v.

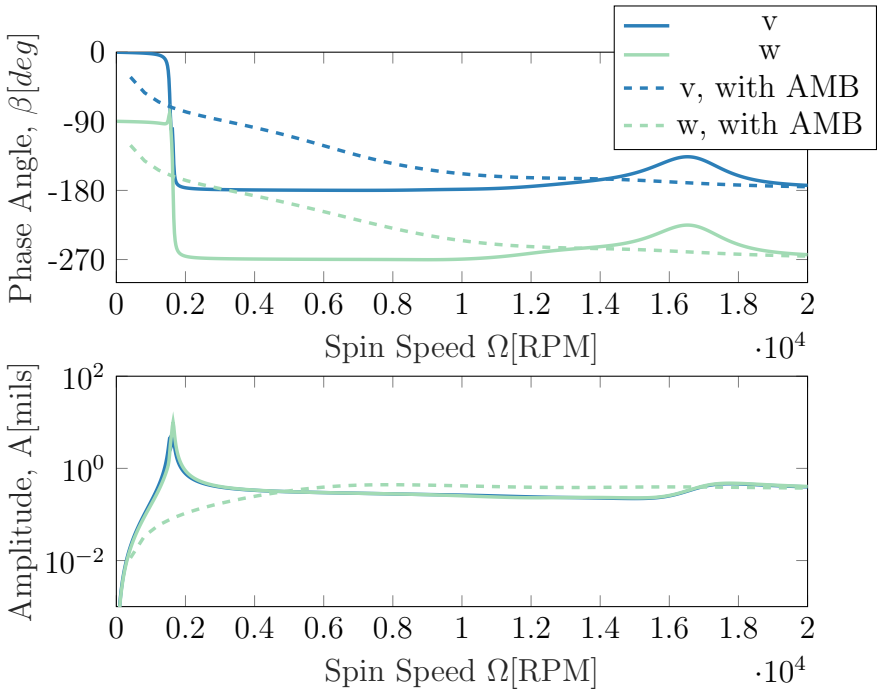


Figure 4.12: Bode diagram at node 6 comparing the rotor without AMB(solid) and with AMB(dashed).

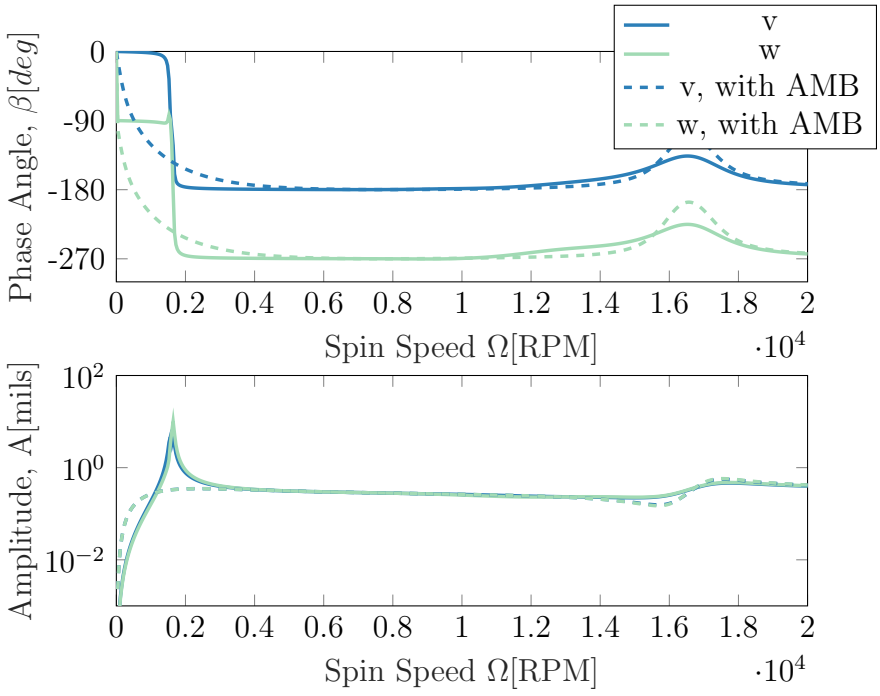


Figure 4.13: Bode diagram at node 6 comparing the rotor without AMB(solid) and with AMB(dashed) for complete levitation at node 4.

Chapter 5

CONCLUSION

5.1 Summary

The aim of this work was to provide a basis for development of rotordynamic models, and their comparison to experimental results. A finite element method designed specifically for creation of these models was presented. Techniques for analyzing models in the frequency domain were explained and the results demonstrated. Methods for also analyzing experimental vibration signals provided a correlation between models and experimental results. Correlations between experimental results and theoretical models were explored in the optimization of an overhung rotor levitated by an active magnetic bearing. Interpretations of both model and experimental signal analysis were provided to lend in understanding results. Resulting analyses and interpretations can be used to identify problems with existing rotating machines as well as aid in the design process of new rotating machines.

5.2 Future Work

This work is intended to be a building block on which future students at Cal Poly may build more advanced models and signal processing techniques in the field of rotordynamics. Possible future projects related to the extension of this work are:

- an experiment with the application of an active magnetic bearing on an overhung rotor system, using the parameters and methods determined in this work
- extension of this finite element model to include non-linear internal damping constitutive relationships

- extension of this finite element model to include a more detailed disk model for the analysis of turbomachines
- extension of this finite element model to include damping effects from various fittings and couplings

BIBLIOGRAPHY

[1] L. Andersen and S. R. Nielsen. Elastic beams in three dimensions. *Aalborg University*, 2008.

[2] D. E. Bently and C. T. Hatch. Fundamentals of rotating machinery diagnostics. *Mechanical Engineering-CIME*, 125(12):53–54, 2003.

[3] D. W. Childs. *Turbomachinery rotordynamics: phenomena, modeling, and analysis*. John Wiley & Sons, 1993.

[4] R. R. Craig and A. J. Kurdila. *Fundamentals of structural dynamics*. John Wiley & Sons, 2006.

[5] A. Das, M. Nighil, J. Dutt, and H. Irretier. Vibration control and stability analysis of rotor-shaft system with electromagnetic exciters. *Mechanism and Machine Theory*, 43(10):1295–1316, 2008.

[6] B. Ertas and J. Vance. The influence of same-sign cross-coupled stiffness on rotordynamics. *Journal of Vibration and Acoustics*, 129(1):24–31, 2007.

[7] L. Forrai. Stability analysis of symmetrical rotor-bearing systems with internal damping using finite element method. In *International Gas Turbine and AeroB engine Congress and Exhibition, Birmingham, UK*, 1996.

[8] G. Genta. Consistent matrices in rotor dynamic. *Meccanica*, 20(3):235–248, 1985.

[9] G. Genta. On a persistent misunderstanding of the role of hysteretic damping in rotordynamics. *TRANSACTIONS-AMERICAN SOCIETY OF MECHANICAL ENGINEERS JOURNAL OF VIBRATION AND ACOUSTICS*, 126(3):459–461, 2004.

[10] G. Genta. *Dynamics of rotating systems*. Springer Science & Business Media, 2007.

[11] T. Gmur and J. Rodrigues. Shaft finite elements for rotor dynamics analysis. *Journal of Vibration and Acoustics*, 113(4):482–493, 1991.

[12] U. Goerguelue. Beam theories the difference between euler-bernoulli and timoschenko. *Lecture Handouts*, 2009.

[13] P. Goldman and A. Muszynska. Application of full spectrum to rotating machinery diagnostics. *Orbit*, 20(1):17–21, 1999.

[14] L. Greenhill, W. Bickford, and H. Nelson. A conical beam finite element for rotor dynamics analysis. *Journal of Vibration, Acoustics, Stress, and Reliability in Design*, 107(4):421–430, 1985.

[15] M. A. Kandil. *On rotor internal damping instability*. PhD thesis, Imperial College London (University of London), 2005.

[16] B. Kirchgäßner. Finite elements in rotordynamics. *Procedia Engineering*, 144:736–750, 2016.

[17] Y. Luo. An efficient 3d timoshenko beam element with consistent shape functions. *Adv. Theor. Appl. Mech*, 1(3):95–106, 2008.

[18] A. Muszynska. *Rotordynamics*. CRC press, 2005.

[19] H. Nelson. A finite rotating shaft element using timoshenko beam theory. *Journal of mechanical design*, 102(4):793–803, 1980.

[20] H. Nelson and J. McVaugh. The dynamics of rotor-bearing systems using finite elements. *Journal of Engineering for Industry*, 98(2):593–600, 1976.

[21] H. N. Ozguven and Z. L. Ozkan. Whirl speeds and unbalance response of multibearing rotors using finite elements. *Journal of vibration, acoustics, stress, and reliability in design*, 106(1):72–79, 1984.

[22] E. Swanson, C. D. Powell, and S. Weissman. A practical review of rotating machinery critical speeds and modes. *Sound and vibration*, 39(5):16–17, 2005.

[23] H. Wettergren and K.-O. Olsson. Dynamic instability of a rotating asymmetric shaft with internal viscous damping supported in anisotropic bearings. *Journal of sound and vibration*, 195(1):75–84, 1996.

[24] E. Zorzi and H. Nelson. Finite element simulation of rotor-bearing systems with internal damping. *Journal of Engineering for Power*, 99(1):71–76, 1977.

APPENDICES

Appendix A

BERNOULLI-EULER BEAM EQUATION

Assumptions used to derive the Bernoulli-Euler beam equation are (Complete derivation in [4]):

1. Beam is bending in a plane, in this case in the y-direction, where the x-direction is along the length of the beam.
2. The neutral axis undergoes no deformation in the longitudinal direction.
3. Cross sections remain plane and perpendicular to the nuetral axis.
4. The material is linear-elastic.
5. Stresses in the y and z direction are negligible compared to those in the x direction.
6. Rotary inertial effects are not considered.
7. Mass density is constant at each cross section, so that each mass center is coincident with the centroid of that section.

Using kinematics and assumptions 2 & 3, the strain in the x direction may be related to the curvature of the beam, $\mu(x,t)$, and the distance from the neutral axis by

$$\epsilon = -\frac{y}{\mu} \tag{A.1}$$

then, with assumption 4 & 7 the relation from curvature to moment is

$$M(x,t) = \frac{EI}{\mu} \tag{A.2}$$

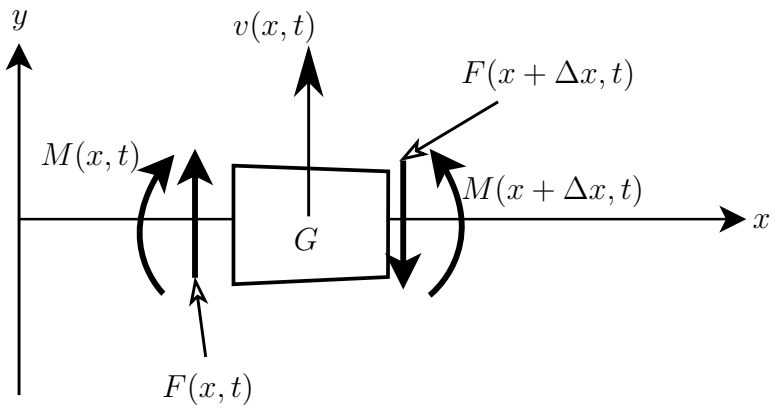


Figure A.1: Free body diagram of a beam section in planar bending.

where E, Young's modulus, and I, area moment of inertia are constant in cross sections. By using Newton's laws and the free body diagram of a single beam element, see Figure A.1, the equations of motion are summarized as:

$$\sum F_y = \Delta m \ddot{v} \quad \& \quad \sum M_G = 0 \tag{A.3}$$

Moment equation is represented as moments summarized at the center of mass, G. The right hand side of moment equation of EOM (A.3) is know to be null due to assumption 6. Applying Newton's equations (A.3) to the FBD of Figure A.1 results in the force equation

$$F(x,t) - F(x + \Delta x, t) = \rho A \Delta x \frac{\partial^2 v}{\partial t^2} \tag{A.4}$$

and moment equation

$$-M(x,t) + M(x + \Delta x, t) + F(x,t)\left(\frac{-\Delta x}{2}\right) + [-F(x + \Delta x, t)]\left(\frac{\Delta x}{2}\right) = 0 \tag{A.5}$$

Taking the limit of equations (A.4) & (A.5) as $\Delta x \to 0$ results in equations (A.6) & (A.7) respectively.

$$\frac{\partial F}{\partial x} = -\rho A \frac{\partial^2 v}{\partial t^2} \tag{A.6}$$

$$\frac{\partial M}{\partial x} - F = 0 \tag{A.7}$$

Assuming the beam slope, $\frac{\partial v}{\partial x}$, remains relatively small, then linearized curvature of the beam is inversely related to $\frac{\partial^2 v}{\partial x^2}$. Substituting this linearized curvature in (A.2) produces

$$M(x,t) = EI\frac{\partial^2 v}{\partial x^2} \tag{A.8}$$

Using linearized moment equation (A.8), combined with (A.6) & (A.7) lends the Euler beam equation

$$\frac{\partial^2}{\partial x^2}\left(EI\frac{\partial^2 v}{\partial x^2}\right) = -\rho A\frac{\partial^2 v}{\partial t^2} \tag{A.9}$$

This is the governing differential equation for transverse motion of a slender beam. This equation is not suitable for an application involving lengths that are not much greater than the width of the beam [10].

Appendix B

ROTORFEM MATLAB CODE FOR CONSTRUCTING AND ANALYZING FEM MODELS

B.1 Main Object File

```
classdef RotorFEM< handle
    %MODEL Summary of this class goes here
    %   Detailed explanation goes here

    properties
        M
        C
        K
        F
        input_file
        npos

    end
    properties (Access = private)

    end
    methods
        function obj = Model(input_filename)
            if nargin == 1
                obj.Assem(input_filename);
                obj.input_file = input_filename;
            end
        end
    end

end
```

B.2 Matrix Assembly File

```
function Assem(obj,input_file)
run(input_file)

nodes = size(elems,1)+1;
npos = zeros(nodes,1);
for ii = 1:nodes-1
    npos(ii+1) = npos(ii) + elems(ii,2);
end
obj.npos = npos;

K = zeros(nodes*DOF,nodes*DOF);
```

```
C = zeros(nodes*DOF,nodes*DOF);
M = zeros(nodes*DOF,nodes*DOF);
F = zeros(nodes*DOF,1);
for ii = 1:1:size(elems,1)
    E = mate(elems(ii,1),2);
    l = elems(ii,2);
    do = elems(ii,3);
    if size(elems(ii,:))<4
        di=0;
    else
        do=elems(ii,4);
    end
    I = obj.AInert(do,di);
    A = obj.Area(do,di);
    rho = mate(elems(ii,1),3);
    nuv = mate(elems(ii,1),4);
    poi = mate(elems(ii,1),5);
    Id = obj.DiaInert(do,l,rho,di);
    Ip = obj.PolInert(do,l,rho,di);
    [kbe, cbe, mbe] = obj.TBeam(E,I,l,A,rho,Id,Ip,nuv,poi,DOF);
    K = obj.Add(K,kbe,(ii-1)*DOF+1:(ii+1)*DOF);
    C = obj.Add(C,cbe,(ii-1)*DOF+1:(ii+1)*DOF);
    M = obj.Add(M,mbe,(ii-1)*DOF+1:(ii+1)*DOF);
end

if exist('disks', 'var')
    for ii = 1:1:size(disks,1)
        do = disks(ii,3);
        l = disks(ii,2);
        rho = mate(disks(ii,1),3);
        a = disks(ii,4);
        Ki = disks(ii,5);
        if size(disks(ii,:))<7
            di=0;
        else
            di=disks(ii,7);
        end
        md = obj.Mass(do,l,rho,di);
        Ip = obj.PolInert(do,l,rho,di);
        Id = obj.DiaInert(do,l,rho,di);
        [mi,gi,fi] = obj.Disk(md,Id,Ip,a,Ki,DOF);
        M = obj.Add(M,mi,(disks(ii,6)*DOF-DOF+1:disks(ii,6)*DOF));
        C = obj.Add(C,gi,(disks(ii,6)*DOF-DOF+1:disks(ii,6)*DOF));
        F = obj.AddVect(F,fi,(disks(ii,6)*DOF-DOF+1:disks(ii,6)*DOF));
    end
end

for ii = 1:1:size(bears,1)
    kx = beartypes(bears(ii,1),2);
    ky = beartypes(bears(ii,1),3);
    c = beartypes(bears(ii,1),4);
    [ki,ci] = obj.Bear(kx,ky,c,DOF);
    K = obj.Add(K,ki,(bears(ii,2)*DOF-DOF+1:bears(ii,2)*DOF));
    C = obj.Add(C,ci,(bears(ii,2)*DOF-DOF+1:bears(ii,2)*DOF));

```

```
end
if exist('mags', 'var')
    for ii = 1:1:size(mags,1)
        olddir = pwd;
        cd(char(magfile(mags(ii,1),2)))
        Magfn = str2func(char(magfile(mags(ii,1),3)));
        [ki, ci] = Magfn();
        cd(olddir)
        K = obj.Add(K,ki,(mags(ii,2)*DOF-DOF+1:mags(ii,2)*DOF));
        C = obj.Add(C,ci,(mags(ii,2)*DOF-DOF+1:mags(ii,2)*DOF));

    end
end
obj.M = M;
obj.C = C;
obj.K = K;
obj.F = F;
```

B.3 Elemental Matrices

```
function [ K,D,M ] = TBeam( obj, E,I,l,A,rho,Id,Ip,nuv,poi,DOF )
%TBeam Timoshenko beam element
% disp1y disp1z ang1y ang1z disp2y disp2z ang2y ang2z disp1x ang1x disp2x ang2x
% ------------------------------------------------------------------------
%|                                                                       |disp1y
%|                                                                       |disp1z
%|                                                                       |ang1y
%|                                                                       |ang1z
%|                                                                       |disp2y
%|                                                                       |disp2z
%|                                                                       |ang2y
%|                                                                       |ang2z
%|-----------------------------------------------------------------------|ang2x
%% Properties %%
k = 6*(1+poi)/(7+6*poi);
G = E/2/(1+poi);
alph = 12*E*I/k/G/A/l^2;
%% Variables %%
syms x
z = x/l;
%% Shape Functions %%
N1 = 1-z;
N2 = z;
Tt1 = (1/(1+alph))*(2*z^3 - 3*z^2 - alph*z + 1 + alph);
Tt2 = (1/(1+alph))*(-2*z^3+3*z^2+alph*z);
Tr1 = (1/(1+alph))*(z^3 - (2+1/2*alph)*z^2 + (1+1/2*alph)*z);
Tr2 = (1/(1+alph))*(z^3 - (1-1/2*alph)*z^2 - 1/2*alph*z);
Rt1 = 6/l*(1/(1+alph))*(z^2-z);
Rt2 = -Rt1;
Rr1 = (1/(1+alph))*(3*z^2 - (4+alph)*z + 1 + alph);
Rr2 = (1/(1+alph))*(3*z^2 - (2-alph)*z);
%% Transformation Matricies %%
```

```
P = [0  0 0 0 0 0;
     0  0 0 0 0 0;
     0 -1 0 0 0 0;
     1  0 0 0 0 0;
     0  0 0 0 0 0;
     0  0 0 0 0 0];
Di = [12*E*I/alph/l^2          0        0   0   0            0;
     0          12*E*I/alph/l^2 0   0   0            0;
     0                  0       E*I 0   0            0;
     0                  0       0   E*I 0            0;
     0                  0       0   0   E*A          0;
     0                  0       0   0   0   12*E*I/alph/l^2/A*2*I];
Mi = diag([rho*A*l,rho*A*l,Id,Id,rho*A*l,Ip])./l;
Gi = kron(diag([0,1,0]),[0, Ip;-Ip,0])./l;
T = kron(diag([1,1,0]),[0,1;-1,0]);
N = [   Tt1   0    0   Tr1 Tt2  0    0   Tr2 0  0  0  0;
     0   Tt1  Tr1  0   0   Tt2  Tr2  0  0  0  0  0;
     0   Rt1  Rr1  0   0   Rt2  Rr2  0  0  0  0  0;
     Rt1   0    0   Rr1 Rt2  0    0   Rr2 0  0  0  0;
     0    0    0   0   0   0    0   0   N1 0  N2 0;
     0    0    0   0   0   0    0   0   0  N1 0  N2];
%% DOF Modifications%%
switch DOF
    case 6
        I = diag([1,1,-1,1,1,1]);
        N = I*N*[I,zeros(6);zeros(6),I]; % Adjust for use of -angy,-angl,2y definitions in shape
            functions
    case 4
        P = P(1:4,1:4);
        Di = Di(1:4,1:4);
        Mi = Mi(1:4,1:4);
        Gi = Gi(1:4,1:4);
        T = T(1:4,1:4);
        I = diag([1,1,-1,1]); % Adjust for use of -angy,-angl,2y definitions in shape functions
        N = I*N(1:4,1:8)*[I,zeros(4);zeros(4),I];
    case 2
        P = P(2:3,2:3);
        Di = Di(2:3,2:3);
        Mi = Mi(2:3,2:3);
        Gi = Gi(2:3,2:3);
        T = T(2:3,2:3);
        N = N(2:3,[2,3,6,7]);
end
%% Elemental Integration %%
B = diff(N.',x) - N.'*P;
B = B.';
K_B = int(B.'*Di*B,x,0,l);
K_C = int(B.'*T*Di*B,x,0,l);
G = int(N.'*Gi*N,x,0,l);
M = int(N.'*Mi*N,x,0,l);
%% Case of symbolic Integration %%
if isa(l,'sym') || isa(E,'sym') || isa(I,'sym') || isa(nuv,'sym')
else
    K_B = double(K_B);
    K_C = double(K_C);
```

```
    G = double(G);
    M = double(M);
end
K =       K_B + nuv.*1i.*K_C;
D = nuv.*K_B +       1i.*G;

end
```

```
function [Md,Gd,Fd] = Disk(obj,md,Id,Ip,a,Ki,DOF)
%% Creates Disk nodal stiffness matrix in the form:
% disp1x disp1y ang1x ang1y
% -------------------------
%|                          |disp1x
%|                          |disp1y
%|                          |ang1x
%|--------------------------|ang1y

%% Begin Function
Fx = md*a;
Fy = md*a;
Mx = (Id-Ip)*Ki;
My = (Id-Ip)*Ki;
Gi=[0,Ip;-Ip,0];
switch DOF
    case 6
        Md = diag([md,md,Id,Id,md,Ip]);
        Gd = 1i*kron(diag([0,1,0]),Gi);
        Fd = [Fx; Fy; Mx; My; 0; 0];
    case 4
        Md = diag([md,md,Id,Id]);
        Gd = 1i*kron(diag([0,1]),Gi);
        Fd = [Fx; Fy; Mx; My];
    case 2
        Md = diag([md,Id]);
        Gd = 1i*[0,0;0,Ip];
        Fd = [Fx; My];
end
% Fx = md*a*(w^2*cos(theta+phi1)-alph*sin(theta+phi1)); %N, Forcing funtion at disk 1
% Fy = md*a*(w^2*sin(theta+phi1)+alph*cos(theta+phi1)); %N, Forcing funtion at disk 2
% Mx = -w^2*(Ip - Id)*Ki*cos(theta+phi2); %N, Forcing funtion at disk 1
% My = w^2*(Ip - Id)*Ki*sin(theta+phi2); %N, Forcing funtion at disk 2;
end
```

```
function [ Kb,Cb ] = Bear(obj,kx,ky,c,DOF)
%% Creates bearing nodal stiffness matrix in the form:
% disp1y disp1z ang1y ang1z disp1x ang1x
% --------------------------------------
%|                                      |disp1y
%|                                      |disp1z
%|                                      |ang1y
%|                                      |ang1z
%|                                      |disp1x
%|--------------------------------------|ang1x
%% Begin function
switch DOF
```

```
    case 6
        Kb = diag([kx,ky,0,0,0,0]);
        Cb = diag([c*kx,c*ky,0,0,0,0]);
    case 4
        Kb = diag([kx,ky,0,0]);
        Cb = diag([c*kx,c*ky,0,0]);
    case 2
        k = mean([kx,ky]);
        Kb = diag([k,k]);
        Cb = diag([c*k,c*k]);
end
end
```

B.4 Plotting Functions

```
function RootLocus(Model_obj,Omega,plotmodes, ax, linetp,color)
%CAMPBELL Summary of this function goes here
%   Detailed explanation goes here
zci = @(v) find(v(:).*circshift(v(:), [-1 0]) <= 0);
if nargin < 6
    color = [0,0,0];
    if nargin <5
    linetp = '.';
    end
    if nargin < 4
        figure
        ax = axes;
    end
    if nargin < 3
        plotmodes = 1:2;
    end
    if nargin < 2
        return
    end

end

for ii = 1:1:length(Omega)
    w = Omega(ii)/60*2*pi;
    Mnew = Model_obj.M;
    Cnew = real(Model_obj.C) + w.*imag(Model_obj.C);
    Knew = real(Model_obj.K) + w.*imag(Model_obj.K);
    Zer = zeros(size(Mnew));
    ey = eye(size(Mnew));
    A = [-Mnew^-1*Cnew, -Mnew^-1*Knew;
        ey,Zer];
    v1=eig(A);
%     AA=[Cnew Mnew;Mnew Zer];
%     B=[Knew Zer;Zer -Mnew];
%     v1=eig(B,-1i*AA);
    [~, I] = (sort(abs((v1))));
    eiv(:,ii) = v1((I));
```

```
end
% while min(abs(imag(eiv(1,:)))) == 0
%     eiv(1:2,:) = [];
% end
% eiv = eiv( ~any( isnan( eiv ) | isinf( eiv ), 2 ),: );

hold on
for jj = 1:1:length(plotmodes)

%     plot(ax,real(eiv(plotmodes(jj)*4-1,:)),(imag(eiv(plotmodes(jj)*4-1,:))),linetp,'Color
    ',[0.8500,0.3250,0.0980])
%     plot(ax,real(eiv(plotmodes(jj)*4-3,:)),(imag(eiv(plotmodes(jj)*4-3,:))),linetp,'Color
    ',[0,0.4470,0.7410])
    plot3(real(eiv(plotmodes(jj)*4-1,:)),abs(imag(eiv(plotmodes(jj)*4,:))),Omega,linetp,'Color',
        color,'MarkerSize',1);%,'o','Color',[0.8500,0.3250,0.0980],'MarkerSize',3)
    plot3(real(eiv(plotmodes(jj)*4-3,:)),abs(imag(eiv(plotmodes(jj)*4-2,:))),Omega,linetp,'Color',
        color,'MarkerSize',1);%,'.','Color',[0,0.4470,0.7410],'MarkerSize',4)
%     plot3(real(eiv(plotmodes(jj)*4-0,:)),imag(eiv(plotmodes(jj)*4-0,:)),Omega,'.','Color
    ',[0,0.4470,0.7410])
%     plot3(real(eiv(plotmodes(jj)*4-2,:)),imag(eiv(plotmodes(jj)*4-2,:)),Omega,'.','Color
    ',[0,0.4470,0.7410])
%     r(1,:) = real(eiv(plotmodes(jj)*4-1,:));
%     r(2,:) = real(eiv(plotmodes(jj)*4-3,:));
%     ZeroCross1 = zci(r(1,:));
%     ZeroCross2 = zci(r(2,:));
%     if isempty(ZeroCross1)
%     else
%     strcross1 = [num2str(Omega(ZeroCross1(1)),4) ' (RPM) ' newline];
%     text(ax,0,abs(imag(eiv(plotmodes(jj)*4-1,ZeroCross1(1)))),strcross1,'HorizontalAlignment','
    right','VerticalAlignment','bottom','FontWeight','bold');
%     plot(ax,0,abs(imag(eiv(plotmodes(jj)*4-1,ZeroCross1(1)))),'*k');
%     end
%     if isempty(ZeroCross2)
%     else
%     strcross2 = [ num2str(Omega(ZeroCross2(1)),4) ' (RPM) ' newline];
%     text(ax,0,abs(imag(eiv(plotmodes(jj)*4-3,ZeroCross2(1)))),strcross2,'HorizontalAlignment','
    right','VerticalAlignment','bottom','FontWeight','bold');
%     plot(ax,0,abs(imag(eiv(plotmodes(jj)*4-3,ZeroCross2(1)))),'*k');
%     end
end
hold off
ax.YAxisLocation = 'origin';
axis([-inf,inf,0,inf])
xlabel('Real(s)')
ylabel('Imag(s)')
end
```

```
function Campbell(Model_obj,Omega,plotmodes,ax,linetp)
%CAMPBELL Summary of this function goes here
%   Detailed explanation goes here
if nargin < 5
    if nargin == 2
        plotmodes = 1:2;
    elseif nargin < 2
        return
```

```
    end
    linetp = '.';
    if nargin < 4
        ax = [];
    end
end
for ii = 1:1:length(Omega)
    w = Omega(ii)/60*2*pi;
    Mnew = Model_obj.M;
    Cnew = real(Model_obj.C) + w.*imag(Model_obj.C);
    Knew = real(Model_obj.K) + w.*imag(Model_obj.K);
    Zer = zeros(size(Mnew));
    ey = eye(size(Mnew));
    A = [-Mnew^-1*Cnew, -Mnew^-1*Knew;
        ey,Zer];
    v1=eig(A);
    [~, I] = sort(abs((v1)));
    eiv(:,ii) = v1(I);
% eiv(:,ii)=v1;
end
if isempty(ax)
    figure
    ax = axes;
end
% while min(abs(imag(eiv(1,:)))) == 0
%     eiv(1:2,:) = [];
% end
% eiv = eiv( ~any( isnan( eiv ) | isinf( eiv ), 2 ),: );

hold on
for jj = 1:1:length(plotmodes)
    plot(ax,Omega,abs(imag(eiv(plotmodes(jj)*4-2,:)))/2/pi*60,'.k');%,'Color',[0,0.4470,0.7410],'
        LineWidth',1)
    plot(ax,Omega,abs(imag(eiv(plotmodes(jj)*4,:)))/2/pi*60,'.k');%, 'Color
        ',[0.8500,0.3250,0.0980],'LineWidth',1,'MarkerSize',4)
%         plot(ax,Omega,(imag(eiv(plotmodes(jj)*4-3,:)))/2/pi*60,linetp,'Color
    ',[0,0.4470,0.7410],'LineWidth',1)
%     plot(ax,Omega,(imag(eiv(plotmodes(jj)*4,:)))/2/pi*60,linetp, 'Color
    ',[0.8500,0.3250,0.0980],'LineWidth',1)
%
%     plot(ax,Omega,imag(eiv(plotmodes(jj),:))/2/pi*60,linetp,'Color',[0,0.4470,0.7410],'LineWidth
    ',1)

end
plot(Omega,Omega,'--k','LineWidth',2)
% plot(Omega,-Omega,'--k','LineWidth',2)
hold off
xlabel('Spin speed [RPM]')
ylabel('Whirl speed [RPM]')
title('Campbell Diagram')


end
```

```
function Stability( Model_obj, Omega,ax,linetp )
```

```
%STABILITY Summary of this function goes here
%   Detailed explanation goes here
zci = @(v) find(v(:).*circshift(v(:), [-1 0]) <= 0);
if nargin < 4
    linetp = '-';
    if nargin < 3
        figure
        ax = axes;
    elseif nargin < 2
        return
    end

end

for ii = 1:1:length(Omega)
    w = Omega(ii)/60*2*pi;
    Mnew = Model_obj.M;
    Cnew = real(Model_obj.C) + w.*imag(Model_obj.C);
    Knew = real(Model_obj.K) + w.*imag(Model_obj.K);

    Zer = zeros(size(Mnew));
    ey = eye(size(Mnew));
    A = [-Mnew^-1*Cnew, -Mnew^-1*Knew;
        ey,Zer];
    v1=eig(A);
    eiv(:,ii) = sort(v1);
end
while min(abs(imag(eiv(1,:)))) == 0
    eiv(1:2,:) = [];
end
eiv = eiv( ~any( isnan( eiv ) | isinf( eiv ), 2 ),: );
hold on
plot(Omega,max(real(eiv)),'LineWidth',1,'LineStyle',linetp);
ZeroCross = Omega(zci(max(real(eiv))));
if ~isempty(ZeroCross)
    strcross = ['  Threshold: ' num2str(ZeroCross(1),4) 'RPM'];
    text(ZeroCross(1),0,strcross,'HorizontalAlignment','left','VerticalAlignment','middle','
        FontWeight','bold','Rotation',90);
    plot(ZeroCross(1),0,'*k'); hold off
end
ax.XAxisLocation = 'origin';
xlabel('Speed (RPM)')
ylabel('Maximum \Re(s)')
title('Stability region')
end
```

```
function Damping(Model_obj,Omega,plotmodes,ax,linetp,ch)
%CAMPBELL Summary of this function goes here
%   Detailed explanation goes here
zci = @(v) find(v(:).*circshift(v(:), [-1 0]) <= 0);

if nargin < 6
    if nargin == 2
        plotmodes = 1:2;
    elseif nargin < 2
```

```
        return
    end
    ch = 'dec';
    if nargin < 5
        linetp = '-';
    end
    if nargin < 4
        figure
        ax = axes;
    end
end

for ii = 1:1:length(Omega)
    w = Omega(ii)/60*2*pi;
    Mnew = Model_obj.M;
    Cnew = real(Model_obj.C) + w.*imag(Model_obj.C);
    Knew = real(Model_obj.K) + w.*imag(Model_obj.K);

    Zer = zeros(size(Mnew));
    ey = eye(size(Mnew));
    A = [-Mnew^-1*Cnew, -Mnew^-1*Knew;
        ey,Zer];
    v1=eig(A);
        [~, I] = (sort(abs((v1))));
    eiv(:,ii) = v1((I));
end

while min(abs(imag(eiv(1,:)))) == 0
    eiv(1:2,:) = [];
end
eiv = eiv( ~any( isnan( eiv ) | isinf( eiv ), 2 ),: );

hold on
for jj = 1:1:length(plotmodes)
    idf = plotmodes(jj)*4-1; %forward mode index location
    idb = idf - 2; %backward mode index location
    zeta = -real(eiv([idb,idf],:))./abs(eiv([idb,idf],:));
%     delta = 2*pi.*zeta./sqrt(1-zeta.^2); %Logarithmic Decriment
%     delta = -2*pi*real(eiv([idb,idf],:))./imag(eiv([idb,idf],:));
    plot(Omega,zeta(1,:),linetp,'Color',[0,0.4470,0.7410],'LineWidth',1);
    plot(Omega,zeta(2,:),linetp,'Color',[0.8500,0.3250,0.0980],'LineWidth',1);
    ZeroCross1 = Omega(zci(zeta(1,:)));
    ZeroCross2 = Omega(zci(zeta(2,:)));
    if isempty(ZeroCross1)
    else
    strcross1 = [num2str(ZeroCross1(1),4) newline];
    text(ZeroCross1(1),0,strcross1,'HorizontalAlignment','left','VerticalAlignment','bottom','
        FontWeight','bold');
    plot(ZeroCross1(1),0,'*k');
    end
    if isempty(ZeroCross2)
    else
    strcross2 = [ num2str(ZeroCross2(1),4) newline];
    text(ZeroCross2(1),0,strcross2,'HorizontalAlignment','left','VerticalAlignment','bottom','
        FontWeight','bold');
```

```
    plot(ZeroCross2(1),0,'*k');
    end

end
hold off
ax.XAxisLocation = 'origin';
xlabel('Speed (RPM)')
ylabel('\zeta')
title('Damping Characteristics')

end
```

```
function Shape(Model_obj,Omega,plotmodes,ax,linetp )
%CAMPBELL Summary of this function goes here
%   Detailed explanation goes here
if nargin < 5
    if nargin < 2
        return
    end
    linetp = '-';
    if nargin < 4
        figure
        ax = axes;
    end
end

syms x;
l=1;
z=x/l;
alph=.25;
Tt1 = (1/(1+alph))*(2*z^3 - 3*z^2 - alph*z + 1 + alph);
Tt2 = (1/(1+alph))*(-2*z^3+3*z^2+alph*z);
Tr1 = (1/(1+alph))*(z^3 - (2+1/2*alph)*z^2 + (1+1/2*alph)*z);
Tr2 = (1/(1+alph))*(z^3 - (1-1/2*alph)*z^2 - 1/2*alph*z);
cpsi = [Tt1  0    0  Tr1  Tt2  0    0  Tr2;
        0  Tt1  -Tr1  0    0  Tt2  -Tr2  0];
% psi = [1 - 3*(x/l)^2 + 2*(x/l)^3, x - 2*l*(x/l)^2 + l*(x/l)^3, 3*(x/l)^2 - 2*(x/l)^3, -l*(x/l)^2
    + l*(x/l)^3];
% cpsi = [psi(1),      0,       0, psi(2), psi(3),      0,       0, psi(4);
%     0, psi(1), -psi(2),      0,       0, psi(3), -psi(4),      0];
ipts = 4;
d = (0:ipts-1)/ipts;
for ii = 1:1:length(d)
    Nx(ii,:) = subs(cpsi(1,:),x,d(ii));
    Ny(ii,:) = subs(cpsi(2,:),x,d(ii));
end
Nx = double(Nx);
Ny = double(Ny);
w = Omega/60*2*pi;
Mnew = Model_obj.M;
Cnew = real(Model_obj.C) + w.*imag(Model_obj.C);
Knew = real(Model_obj.K) + w.*imag(Model_obj.K);
nM = length(Mnew);

% Zer = zeros(nM);
```

```
% AA=[Cnew Mnew;Mnew Zer];
% B=[Knew Zer;Zer -Mnew];
    Zer = zeros(size(Mnew));
    ey = eye(size(Mnew));
    A = [-Mnew^-1*Cnew, -Mnew^-1*Knew;
        ey,Zer];

[V,D] = eig(A,'vector');
% ind=find( (imag(D) > 0) & (abs(D)>1e-3) );    % take positive frequencies only
% l1=D(ind);
% v1=V(nM+1:end,ind);
l1=D;
v1=V(nM+1:end,:);
% v1=V(1:nM,:);
[~, I] = sort(abs((l1)));
l2 = D(I);
v2 = V(:,I);
v = v2(:,plotmodes);
v=v/norm(v);
for ii = 1:1:length(Model_obj.M)/4-1
    Li = Model_obj.npos(ii+1)-Model_obj.npos(ii);
    Nxnew = Nx*diag([1 1 1 Li 1 1 1 Li]);
    Nynew = Ny*diag([1 1 Li 1 1 1 Li 1]);
    pos(ipts*(ii-1)+1:ipts*(ii-1)+ipts+1) = linspace(Model_obj.npos(ii),Model_obj.npos(ii+1),ipts
        +1);
    pos(end)=[];
    shapex(ipts*(ii-1)+1:ipts*(ii-1)+ipts) = Nxnew*v(4*(ii-1)+1:4*(ii-1)+8);
    shapey(ipts*(ii-1)+1:ipts*(ii-1)+ipts) = Nynew*v(4*(ii-1)+1:4*(ii-1)+8);
end
pos = [pos, Model_obj.npos(end)];
shapex = [shapex, v(nM-3)];
shapey = [shapey, v(nM-2)];
plot(ax,[pos;pos]',real([shapex; shapey])')
hold on
for ii = 1:length(Model_obj.npos)
    plot([Model_obj.npos(ii);Model_obj.npos(ii)]',real([shapex(ipts*(ii-1)+1); shapey(ipts*(ii-1)
        +1)])','.k','MarkerSize',8)
end
hold off
figure
cpts = 20;
theta = linspace(0,2*pi,cpts);
for ii =1:length(shapex)
    xx(:,ii)=real(shapex(ii))*cos(theta) - imag(shapex(ii))*sin(theta);
    y(:,ii)=real(shapey(ii))*cos(theta) - imag(shapey(ii))*sin(theta);
end
N = repmat(pos,cpts,1);
hold on
plot3(real(shapex),pos,real(shapey),'-k')

plot3(xx,N,y,'Color',[0,0.4470,0.7410])

for ii = 1:length(Model_obj.npos)
    plot3(real(shapex(ipts*(ii-1)+1)),Model_obj.npos(ii),real(shapey(ipts*(ii-1)+1)),'.k','
        MarkerSize',8)
```

```
end

hold off
mx = max([abs(shapex),abs(shapey)]);
axis([-mx, mx, -inf, inf,-mx, mx ]);
disp(['Damping: ' num2str(real(l2(plotmodes))) '. Frequency: ' num2str(imag(l2(plotmodes))/2/pi
    *60) '(RPM).'])
ax=gca;
ax.XDir='reverse';
ax.Title.String={(['\Re(s): ' num2str(real(l2(plotmodes)),2) ', \Im(s): ' num2str(imag(l2(
    plotmodes))/2/pi*60,4) '[RPM]']);(['\Omega: ' num2str(Omega,4) '[RPM]'])};
%
% [V, D] = eig(Knew, Mnew,'vector');
% D = sqrt(D)*60/2/pi;
% [D, ind] = sort(abs(D));
% D(plotmodes)
% V = V(:,ind);
% v2 = (V(:,plotmodes));
% v3 = reshape(v2,4,[]);
% v4 = bsxfun(@times, v3, 1./sqrt(sum(v3.^2, 2)));
% v2 = reshape(v4,1,[]) ';
% for ii = 1:1:length(Model_obj.M)/4-1
%     shape(8*(ii-1)+1:8*(ii-1)+8) = N*v2(4*(ii-1)+1:4*(ii-1)+8);
% end
% shape = [shape, v2(end-3)];
% plot(real(shape))


% if isempty(ax)
%     figure
%     ax = axes;
% end
%
% hold on
% for jj = 1:1:length(plotmodes)
%     pv(1,:) = real(eivect(speed,[4.*(1:14)-3],plotmodes(jj)));
%
%     plot(pv)
% end
% hold off


end
```

```
function FreqResponse(Model_obj, Omega, node, plottype, fi, linetp )
if nargin < 6
    if nargin == 2
        node = 1;
    elseif nargin < 2
        error('Not enough input arguments.\n Provide: Omega(required), node #s(optional), axes(
            optional), linetype(optional)',class(Model_obj))
    end
    linetp = '-';
    if nargin < 5
        fi = figure;
```

```
        ax = axes(fi);
        if nargin < 4
            plottype = 'Bode';
        end
        if strcmp(plottype,'Bode') == 1
            ax(1)=subplot(2,1,1,ax);
            ax(2)=subplot(2,1,2);
        end
    end

end

for ii = 1:1:length(Omega)
    w = Omega(ii)/60*2*pi;
    Fnew = w^2.*(Model_obj.F);
    Mnew = Model_obj.M;
    Cnew = real(Model_obj.C) + w.*imag(Model_obj.C);
    Knew = real(Model_obj.K) + w.*imag(Model_obj.K);
    n=length(Mnew);
    I=kron(eye(n/2),diag([1,1i]));
    Z = I*(Knew + 1i*w.*Cnew - w^2.*Mnew);
    X(ii,:) = Z^-1*Fnew;
end
switch plottype
    case 'FreqResponse'
        hold on
        for jj = 1:1:length(node)
            ax=fi.Children;
            plot(ax, Omega,(abs(X(:,node(jj)*4-3))),linetp)
            plot(ax, Omega,(abs(X(:,node(jj)*4-2))),linetp)
            ax.YScale = 'log';
        end
        hold off
        ax.XLabel.String='Spin Speed \Omega[RPM]';
        ax.YLabel.String='Amplitude[m]';
    case 'Bode'
        hold on
        for jj = 1:1:length(node)
            subplot(211); hold on
            ax(1)=fi.Children(1);
%             plot(Omega,abs(X(:,node(jj)*4-3)),linetp)
%             plot(Omega,abs(X(:,node(jj)*4-2)),linetp)
            plot(ax(1),Omega,abs(39370.1*X(:,node(jj)*4-3)),linetp)
            plot(ax(1),Omega,abs(39370.1*X(:,node(jj)*4-2)),linetp)
            ax(1).YScale = 'log';
            ax(1).XLabel.String='Spin Speed \Omega[RPM]';
%             ax(1).YLabel.String='Amplitude[m]';
            ax(1).YLabel.String='Amplitude[mils]';
            subplot(212); hold on
            ax(2)=fi.Children(2);
%             plot(Omega,unwrap(mod(angle(X(:,node(jj)*4-3)),2*pi)),linetp)
%             plot(Omega,unwrap(mod(angle(X(:,node(jj)*4-2)),2*pi)),linetp)
            plot(ax(2),Omega,180/pi*unwrap(mod(angle(X(:,node(jj)*4-3)),2*pi)),linetp)
            plot(ax(2),Omega,180/pi*unwrap(mod(angle(X(:,node(jj)*4-2)),2*pi)),linetp)
            ax(2).XLabel.String='Spin Speed \Omega[RPM]';
```

```
66 %                ax.YLabel.String='Phase Angle[Rad]';
67                  ax(2).YLabel.String='Phase Angle[deg.]';
68 %                ax.YTick=[-3*pi/2:pi/2:0];
69 %                ax.YTickLabel={'-3\pi/2','-\pi','-\pi/2','0'};
70                  ax(2).YTick=[-360:90:0];
71                  ax(2).YTickLabel={'0','-270','-180','-90','0'};
72              end
73              hold off
74
75
76  end
77  % plot(Omega,max(abs(X)'))
78  % ax = gca;
79  % ax.YScale = 'log';
```

Appendix C

ROTORDAQ MATLAB CODE FOR ANALYZING VIBRATION SIGNALS

C.1 Main Object File

```
classdef RotorData < handle
    properties
        X %Horizontal Signal
        Y %Vertical Signal
        ref %Keyphasor Signal
        Fs %Sampling Rate (Samp/Sec)
        Fres %Frequency Resolution
        n %filter synchronous multiplier: 1X, 2X, 3X, ... nX the running speed
    end
    properties (SetAccess = private)
        Zf %Total Amplitude
        Phase %Phase
        Amp %Property that holds X and Y amplitudes
        Speed %RPM
    end
    properties (Access = private)
        refchop %%%%%%%%%%%
        xchop   %% Organizes the data arrays into matrices based on windows
        ychop   %%%%%%%%%%%
    end
    properties (Dependent, Access = private)
        NW %Number of windows
        freq %Frequency
        nspw %Number of samples per window

    end
    methods
        function obj = RotorData(x, y, r, f, fres)%Fills properties with data
            if nargin == 5
                obj.X = x;
                obj.Y = y;
                obj.ref = r;
                obj.Fs = f;
                obj.Fres = fres;
            end
        end
        function set.X(obj, val)
            obj.X = val;
        end
        function set.Y(obj, val)
            obj.Y = val;
        end
        function set.ref(obj, val)
            obj.ref = val;
        end
```

```
46          function set.Fs(obj, val)
47              obj.Fs = val;
48              if isempty(obj.X) || isempty(obj.Y) || isempty(obj.ref) || isempty(obj.Fs) || isempty(
                    obj.Fres)
49              else
50                  obj.update;
51              end
52          end
53          function set.Fres(obj, val)
54              obj.Fres = val;
55              if isempty(obj.X) || isempty(obj.Y) || isempty(obj.ref) || isempty(obj.Fs) || isempty(
                    obj.Fres)
56              else
57                  obj.update;
58              end
59          end
60          function set.n(obj, val)
61              obj.n = val;
62              if isempty(obj.X) || isempty(obj.Y) || isempty(obj.ref) || isempty(obj.Fs) || isempty(
                    obj.Fres)
63              else
64                  obj.update;
65              end
66          end
67          function nspw = get.nspw(obj)
68              nspw = ceil(obj.Fs/obj.Fres);
69          end
70          function freq = get.freq(obj)
71              df = obj.Fs/obj.nspw;
72              f = df*(0:obj.nspw-1);
73              Q = ceil((obj.nspw+1)/2); % M/2+1 for M even
74              fQ = df*(Q-1);
75              freq = f-fQ;
76          end
77          function NW = get.NW(obj)
78              NW = floor(length(obj.X)/obj.nspw); %Rounds number of windows down to closest integer
79          end
80          function filter(obj)
81              disp('filtering')
82              for i = 1:1:obj.NW
83                  Fpass1 = obj.Speed(i)*obj.n/60-5; %Upper frequency of bandpass filter
84                  Fpass2 = obj.Speed(i)*obj.n/60+5; %Lower frequency of bandpass filter
85                  if i == 1
86                      h  = fdesign.bandpass('N,Fp1,Fp2,Ast1,Ap,Ast2', 10, Fpass1, Fpass2, 50, .1,
                            50, obj.Fs);
87                      Hd(i) = design(h, 'ellip');
88                      Hd(i).persistentmemory = true;
89                      obj.xchop(:,i) = filter(Hd(i),obj.xchop(:,i)); %Apply filter to data
90                      xf = Hd(i).states;
91                      reset(Hd(i));
92                      obj.ychop(:,i) = filter(Hd(i),obj.ychop(:,i)); %||
93                      yf = Hd(i).states;
94                  else
95                      h  = fdesign.bandpass('N,Fp1,Fp2,Ast1,Ap,Ast2', 10, Fpass1, Fpass2, 50, .1,
                            50, obj.Fs);
```

```
96                      Hd(i) = design(h, 'ellip');
97                      Hd(i).persistentmemory = true;
98                      Hd(i).states = xf;
99                      obj.xchop(:,i) = filter(Hd(i),obj.xchop(:,i)); %Apply filter to data
100                     xf = Hd(i).states;
101                     Hd(i).states = yf;
102                     obj.ychop(:,i) = filter(Hd(i),obj.ychop(:,i)); %||
103                     yf = Hd(i).states;
104                 end
105             end
106         end
107         function update(obj)
108             obj.xchop = zeros(obj.nspw,obj.NW);
109             obj.ychop = zeros(obj.nspw,obj.NW);
110             obj.refchop = zeros(obj.nspw,obj.NW);
111             obj.Speed = zeros(obj.NW,1);
112             x = obj.X(1:obj.nspw*obj.NW);
113             y = obj.Y(1:obj.nspw*obj.NW);
114             r = obj.ref(1:obj.nspw*obj.NW);
115             obj.xchop = reshape(x,obj.nspw,obj.NW);
116             obj.ychop = reshape(y,obj.nspw,obj.NW);
117             obj.refchop = reshape(r,obj.nspw,obj.NW);
118             for i = 1:1:obj.NW
119                 pp = pulseperiod(obj.refchop(:,i),obj.Fs,'Tolerance',7);
120                 obj.Speed(i) = 1./mean(pp)*60;
121             end
122             if ~isempty(obj.n) && ~obj.n == 0
123                 filter(obj)
124             end
125
126             obj.Amp.XAmp = max(obj.xchop) - min(obj.xchop); %Calculates Amplitude by subtracting
                    min and max from each column
127             obj.Amp.YAmp = max(obj.ychop) - min(obj.ychop); %Calculates Amplitude by subtracting
                    min and max from each column
128             Zwin = hanning(obj.nspw, 'Periodic');
129             Z = obj.xchop + 1i*obj.ychop;
130             for i = 1:1:obj.NW
131                 Z(:,i) = Zwin.*Z(:,i);
132             end
133             obj.Zf = 2*abs(fft(Z)/obj.nspw);
134             obj.Zf = fftshift(obj.Zf',2); %Shifts graph to center on the x axis
135
136             obj.Phase.PhaseX = zeros(obj.NW,1);
137             obj.Phase.PhaseY = zeros(obj.NW,1);
138             distance = .5*obj.Fs/100;
139             threshold = .5*(max(obj.refchop(:,ceil(end/2))) - mean(obj.refchop(:,ceil(end/2))));
140             for i = 1:1:obj.NW
141                 [~, locsref] = findpeaks(obj.refchop(:,i),'MinPeakHeight',threshold,'
                        MinPeakDistance',distance);
142                 [~, locsX] = findpeaks(obj.xchop(:,i),'MinPeakProminence',1,'MinPeakDistance',
                        distance);
143                 [~, locsY] = findpeaks(obj.ychop(:,i),'MinPeakProminence',1,'MinPeakDistance',
                        distance);
144                 if length(locsref) < 2 || length(locsX) < 2 || length(locsY) < 2
145                     obj.Phase.PhaseX(i) = NaN;
```

```
146                     obj.Phase.PhaseY(i) = NaN;
147                 else
148                     if length(locsX) ~= length(locsY)
149                         if length(locsX) < length(locsY)
150                             if abs(locsX(1) - locsY(1)) < abs(locsX(end) - locsY(end))
151                                 locsY(end) = [];
152                             else
153                                 locsY(1) = [];
154                             end
155                         else
156                             if abs(locsX(1) - locsY(1)) < abs(locsX(end) - locsY(end))
157                                 locsX(end) = [];
158                             else
159                                 locsX(1) = [];
160                             end
161                         end
162                     end
163                     if locsref(1) > locsX(1) || locsref(1) > locsY(1)
164                         locsX(1) = [];
165                         locsY(1) = [];
166                     end
167                     if length(locsref) > length(locsX)
168                         locsref(length(locsX):end) = [];
169                     end
170                     phx = zeros(length(locsref)-1,1);
171                     phy = zeros(length(locsref)-1,1);
172                     for j = 1:1:length(locsref)-1
173                         phx(j) = mod((locsX(j) - locsref(j))/(locsref(j+1) - locsref(j))*2*pi,2*pi
                                ));
174                         phy(j) = mod((locsY(j) - locsref(j))/(locsref(j+1) - locsref(j))*2*pi,2*pi
                                ));
175                     end
176                     obj.Phase.PhaseX(i) = 180/pi*mod(mean(phx(:)),2*pi);
177                     obj.Phase.PhaseY(i) = 180/pi*mod(mean(phy(:)),2*pi);
178                     clear locsref locsX locsY j
179                 end
180             end
181
182         end
183         function acqListener(obj, src, event)
184             obj.newData = [event.TimeStamps, event.Data]';
185             obj.ref = [obj.ref, event.Data(1,:)];
186             obj.X = [obj.X, event.Data(2,:)];
187             obj.Y = [obj.Y, event.Data(3,:)];
188             obj.update;
189
190         end
191         %Plotting functions below
192         bode(obj)
193         cascade(obj)
194         obj = downsample(obj,r)
195         orbit3(obj)
196         orbitAnimation(obj)
197         mainplot(obj)
198     end
```

```
end
```

C.2 Plotting Functions

```
function bode(obj)
figure('Name','Bode Plot')
subplot(211)
plot(obj.Speed,obj.Amp.XAmp,obj.Speed,obj.Amp.YAmp)
xlabel('Running Speed (RPM)');
ylabel('Amplitude (mils)');
legend('Horizontal Plane','Vertical Plane');
subplot(212)
plot(obj.Speed,obj.Phase.PhaseX,obj.Speed,obj.Phase.PhaseY)
xlabel('Running Speed (RPM)');
ylabel('Phase lag (deg.)');
set(gca,'ytick',-720:45:720,'YTickLabel',{'0','45','90','135','180','225','270','315'})
end
```

```
function cascade(obj)
figure('name','Cascade Plot')
waterfall(obj.freq, obj.Speed, obj.Zf)
xlabel('Frequency of Vibration (Hz)');
ylabel('Running Speed (RPM)');
zlabel('Amplitude of Vibration (mils)');
axis([-100 100 -inf inf 0 inf])
end
```

```

function orbit3(obj)
N = length(obj.X(:));
F = (obj.Speed(end) - obj.Speed(1))/N*(0:N-1) + obj.Speed(1);
s = fix(sqrt(N));
Xlin = obj.X(1:s^2);
Ylin = obj.Y(1:s^2);
w = F(1:s^2);
x = reshape(Xlin,s,s);
y = reshape(Ylin,s,s);
o = reshape(w,s,s);
C = sqrt(x.^2+y.^2);
h = mesh(o,x,y,C);
axis([1000 2000 -inf inf -inf inf]);
xlabel('Running Speed (RPM)');
ylabel('Horizontal Amplitude (mils)');
zlabel('Vertical Amplitude (mils)');
h.FaceColor = 'none';
h.MeshStyle = 'column';
end
```

```
function orbitAnimation(obj)

distance = .5*obj.Fs/100;
```

```
threshold = .5*(max(obj.refchop(:,ceil(end/2))) - mean(obj.refchop(:,ceil(end/2))));
[~, loc] = findpeaks(obj.ref,'MinPeakHeight',threshold,'MinPeakDistance',distance);
pksx=obj.X(loc);
pksy=obj.Y(loc);
Fsh = obj.Fs;
maxlm = max(obj.X);
speedarray = (obj.Speed(end)-obj.Speed(1))/length(pksy)*(0:length(pksy)-1)+obj.Speed(1);
for i = 1:1:length(pksx)-1;
    plot(obj.X(loc(i):loc(i+1)-fix((1/10)*(loc(i+1)-loc(i)))),obj.Y(loc(i):loc(i+1)-fix((1/10)*(
        loc(i+1)-loc(i)))),'-k',pksx(i),pksy(i),'.','markersize',15)
    text = [' Running Speed: ',num2str(fix(speedarray(i)),4),' (RPM)'];
    uicontrol('Style', 'text',...
        'String',text,...
        'Units','normalized',...
        'Position', [.25 .9 0.5 .05]);
    axis([-maxlm maxlm -maxlm maxlm]);
    axis square
    xlabel('Horizontal Amplitude (mils)');
    ylabel('Vertical Amplitude (mils)');
%     pause(1/Fsh)
    pause

end
end
```