

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
Professor of Mechanical Engineering

COMMITTEE MEMBER: Hemanth Porumamilla
Professor of Mechanical Engineering

COMMITTEE MEMBER: Peter Schuster
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, 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 complex state space equations are used to perform stability analysis, produce critical speed maps, and visualize mode shapes. Frequency domain analysis of experimental and theoretical data is used to provide Bode diagrams and full spectrum cascade plots. Synthesis of experimental and theoretical model analysis 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 Introduction	1
1.1 Background	1
1.2 Literature Review	2
1.3 Scope of Work	4
2 Vibration Signal Analysis	5
2.1 Data Collection and Processing	5
2.1.1 Amplitude	8
2.1.2 Spectrum	8
2.1.3 Phase	10
2.1.3.1 Time Domain Approach	10
2.1.3.2 Frequency Domain Approach	11
2.2 Rotordynamic Figures	11
2.2.1 Bode	12
2.2.2 Full Spectrum and Full Spectrum Cascade	12
2.2.3 Orbit	14
2.2.4 Filtering	16
3 Finite Element Method For Rotordynamic systems	19
3.1 Beam Element FE Equation	20
3.1.1 Timoshenko Beam Finite Element	20
3.1.1.1 Kinematic Relationships	21
3.1.1.2 Internal Constitutive Relationship	23
3.1.1.3 Differential Equations of Motion	26
3.1.1.4 Shape Functions	29
3.1.1.5 Finite Equations of Motion	34
3.1.1.6 Rotating Internal Damping	34

3.1.1.7	Beam Element in Complex Coordinates	37
3.2	Disk Nodal Equations	38
3.2.1	Disk in complex coordinates	40
3.3	Bearing Nodal Equations	40
3.3.1	Bearing in Complex Coordinates	41
3.4	Assembly of the Global Systems of Equations	41
3.4.1	In the Real Coordinate System	42
3.4.2	In the Complex Coordinate System	42
3.5	Analysis of the Resulting Model	42
4	Frequency Domain Analysis	44
4.1	State Space Representation and the Eigenvalue Problem	45
4.2	Dynamic Response	47
4.3	Roots Locus and Stability Analysis	50
4.4	Campbell	53
4.5	Shapes	54
5	Synthesis in Example of a Magnetic Bearing on an Overhung Rotor	58
5.1	Physical System Description	58
5.2	Experimental Results	59
5.3	Theoretical Model	61
5.3.1	Active Magnetic Bearing	64
5.3.1.1	Proportional Derivative Control	66
5.4	Addition of Magnetic Bearing to the Rotor Model	68
	BIBLIOGRAPHY	72
	APPENDICES	
A	Bernoulli-Euler Beam equation	76
B	RotorFEM MATLAB Code for Constructing and Analyzing FEM Models	79
B.1	Main Object File	79
B.2	Matrix Assembly File	79
B.3	Elemental Matrices	81
B.4	Plotting Functions	84
C	RotorDAQ MATLAB Code for Analyzing Vibration Signals	94
C.1	Main Object File	94

C.2 Plotting Functions	98
----------------------------------	----

LIST OF TABLES

Table	Page
4.1 Properties of disks, shaft elements, and bearings of the example problem.	46
5.1 Geometric parameters of the overhung rotor system.	59
5.2 Properties of disks, shaft elements, and bearings of the theoretical model.	61
5.3 Active Magnetic Bearing Parameters.	70

LIST OF FIGURES

Figure	Page
2.1 Position of the rotor shaft over a certain timespan.	6
2.2 Rotor Spin Speed Windowing effect.	7
2.3 A window in time of the transient vibration signals for a window, $nspw$, of 2048[<i>samples</i>].	8
2.4 Bode diagram of the experimental example overhung system. Signals are filtered to synchronous speed.	13
2.5 Example Full Spectrum.	14
2.6 Cascade of the experimental system described in §2.2.	15
2.7 3D Orbit of the experimental system described in §2.2. Lighter colors indicate larger vibration.	16
2.8 Orbits of the experimental example. Spin speed is counterclockwise. Blue dots indicate the reference position of the shaft, and the beginning of each orbit.	17
2.9 Cascade of the experimental system with a synchronous filter applied.	18
3.1 Beam Element with nodal displacements.	20
3.2 Timoshenko beam section with degrees of freedom at some point x along beam axis.	21
3.3 Beam differential element with generalized forces.	26
3.4 Beam Element with nodal displacements.	30
3.5 Shape Functions as they vary with ζ using two different ratios of length to radius of beam element.	33
3.6 Depiction of skew angle χ	39
4.1 Diagram of Two disk model example problem.	45
4.2 Bode diagram of the second disk subject to an unbalance at the second disk. Amplitudes of v & w are identical for this antisymmetric system, phase of w would be lagging v by 90[deg].	48
4.3 Nyquist plots for the first two modes at node 6 for the example two disk problem.	49

4.4	Nyquist Diagram for the first mode at node 6 in the whirl speed range $1100 < \omega < 1300[\text{RPM}]$ at a spin speed, $\Omega = 4000[\text{RPM}]$. Counter-clockwise path indicates instability.	49
4.5	Roots Locus of the example problem with an internal damping coefficient of $0.0002[\text{s}]$	51
4.6	Roots Locus of the example problem in 3-D with an internal damping coefficient of $0.0002[\text{s}]$	52
4.7	Stability plot of the example problem.	53
4.8	Damping ratio of the first three modes of the example problem, with indication of threshold of stability for each mode.	53
4.9	Campbell Diagram of the example problem.	54
4.10	Modal shapes of the example problem.	56
5.1	Overhung rotor system diagram.	58
5.2	3D Orbit of the experimental overhung rotor system.	59
5.3	Cascade of the experimental overhung rotor system.	60
5.4	Bode diagram of the experimental overhung rotor filtered to 1X.	60
5.6	Bode Diagram of overhung system without AMB.	63
5.7	Campbell Diagram of the overhung system without AMB.	63
5.8	Roots Locus of overhung system without AMB.	64
5.11	Roots locus of Overhung rotor system with varying k_v	70
5.12	Bode diagram at node 6 comparing the rotor without AMB(solid) and with AMB(dashed).	71
5.13	Bode diagram at node 6 comparing the rotor without AMB(solid) and with AMB(dashed) for complete levitation at node 4.	71
A.1	Free body diagram of a beam section in planar bending.	77

Chapter 1

INTRODUCTION

1.1 Background

The study of rotating machines is a relatively new study of physics beginning in the late 1800 with the industrial revolution driving to the forefront a need to mitigate dangerous operating conditions of machines such as the steam turbine. Increasing demands on power generation were leading to increased size and speed of such turbines, which in turn increased the need for understanding of the vibrations they produced. Engineers and physicists alike thought it was impossible to operate continuously at a speed above the first natural frequency. One of the first papers published on the matter by Rankine in 1869 even proposed this to be fact, that rotors could not be stable above this first natural frequency. Jeffcott produced one of the first meaningful solutions to the rotordynamic problem in 1919 with his model that is now known as the Jeffcott rotor. This rotor is a simple model with a single disk located equidistant from two bearing supports. In his paper he made the realization that after the first natural frequency, a rotor in this configuration is self centering (begins to rotate about its center of mass). This realization led to the operation of many machines in the so-called supercritical speed range, and increased power and efficiency of many industrial machines.

Today our understanding of rotordynamics has led to the operation of some machines for tooling, for example, that spin at speeds of over a hundred thousand revolutions per minute—many times the first natural frequency. The increase in use of computers in the study of rotordynamics has led many innovations allowing new rotors to push these boundaries. Additionally, increasing use of active control devices

such as the Active Magnetic Bearing (AMB) allows for adaptive control of rotating machines.

Rotors being developed today are becoming increasingly complex, as our understanding is allowing them to become. The trend with new systems is driving the analysis of rotordynamics as part of the design process instead of balancing an already built system. The Finite Element Method (FEM) which is widely used in many other fields such as structures and impact analysis, has spread in its use for rotordynamic analysis as part of design. With FEM, the design geometry can be parametrized and systems can be optimized in terms of geometry and material use to minimize vibration. Though readily available FEM packages can be altered for use in rotordynamics, it is often the case that elements available are not designed with the inclusion of gyroscopic moments. Gyroscopic moments provide a vital part aspect of the physics of rotordynamic systems and even more so in high speed systems.

There exists an array of FEM packages to specifically address the rotordynamic model such as: DyRoBeS, DYNROT, and XLRotor. These packages are able to provide valuable design information on complex linear and non-linear FEM models. There also exists many solutions for experimental analysis and monitoring of rotating machines. Bently Nevada has been the leading producer of such monitoring equipment for rotordynamics for several decades, with software and hardware solutions that produce monitoring, diagnostics, and visualizations of rotor vibrations.

1.2 Literature Review

Nelson & McVaugh [30] introduced an early Finite Element for use in rotordynamics that included gyroscopic moments in 1976. This model used a Bernoulli-Euler beam to define the internal bending stiffness of the element, with cubic spline shape interpolation functions. The Bernoulli-Euler beam does not include shear effects limiting

its use to relatively slender beams. Nelson made an improvement to this model in 1977 with Zorzi [39] with the inclusion of internal damping in the constitutive relationship in the form of viscous and hysteretic damping mechanism. Nelson (1980)[29] then extended his 1976 model to include shear deformation of the beam element, known as the Timoshenko beam element. But, did not include internal viscous or hysteretic damping in the model. Ozguven, in 1984[31], extended Zorzi's 1980 Timoshenko model to include shear deformation effects. An excellent derivation of the Timoshenko beam element is provided by Luo (2008)[26]. His derivation results in the static homogeneous Euler-Lagrangian differential equations that pertain to bending, axial, and torsional motion. Additionally, Luo provides a derivation of the shape interpolation functions that reduce shear locking in the Timoshenko beam element. All of these beam models assume a homogeneous beam for each element. Derivations of beam models exist that do not assume a homogeneous beam such as the paper on conical beam elements by Greenhill, Bickford, and Nelson in 1985[20], and the paper on consistent beams from Genta in 1985[14]. This non-homogeneous extension is not realized in the model presented in this work. Genta's textbook on rotordynamics, published in 2007[16] involves a detailed derivation of the finite element equations using the Timoshenko beam and shear-lock reducing shape interpolation functions.

Important anecdotal notes on understanding of the rotating damping effect are presented in Kandil's 2004[23] thesis "On Rotor Internal Damping Instability." Kandil presents a strong physical description of the internal damping effect and the many sources it has. Further interpretations for the effect of internal damping were presented by Genta in his paper "On a Persistent Misunderstanding of the Role of Hysteretic Damping in Rotordynamics." Where the common misunderstanding of sub-critical instability due to hysteretic damping is rectified.

The use of Bernoulli-Euler beam elements in the application of a vibration reducing Active Magnetic Bearing (AMB) is presented in the paper from Das, et al.

(2008[9]). Das also provides the derivation of a simple control algorithm for the AMB using a linearized magnetic force equation.

1.3 Scope of Work

This work will be comprised

Chapter 2

VIBRATION SIGNAL ANALYSIS

2.1 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 beneficial 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 rotation speed of the shaft. It is important to note that these signals can come from an experiment or a theoretical model.

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, $A[m, \text{mils}]$, Amplitude Spectrum, $\tilde{A}(\omega)[m, \text{mils}]$, Phase of Vibration, $\beta[\text{deg}, \text{Rad}]$, and Spin Speed, $\Omega[\text{Hz}, \text{Rad/s}, \text{RPM}]$. Amplitude Spectrum is actually a two dimensional variable where the dependent variable, $\omega[\text{Hz}, \text{Rad/s}, \text{RPM}]$, represents the Frequency of Vibration (or 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 $\tilde{A}(\omega)$. 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

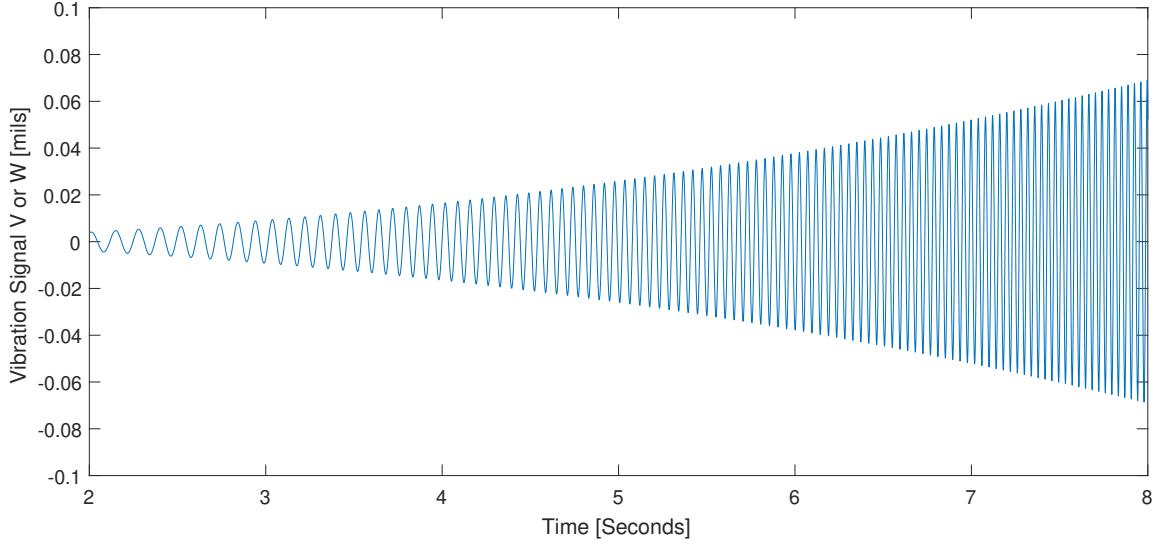
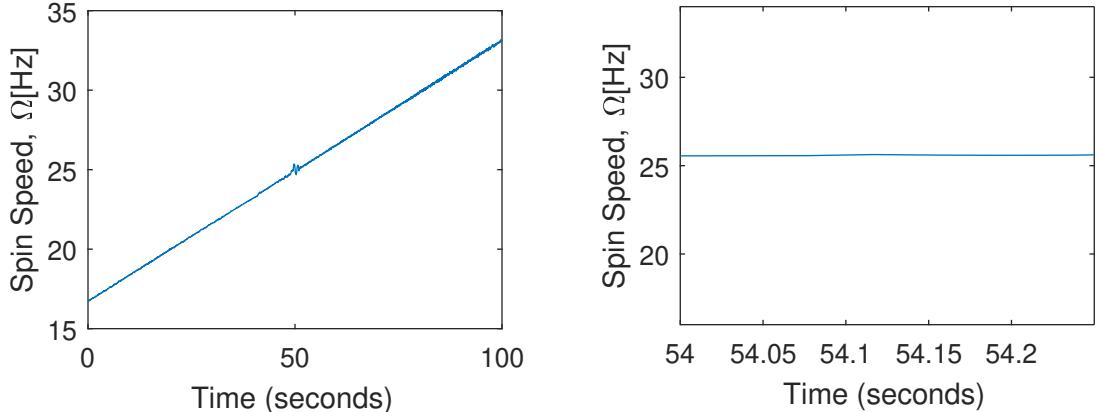


Figure 2.1: Position of the rotor shaft over a certain timespan.

in producing these plots are to follow.

A difficulty in rotordynamic analysis arises in the continuous change of Ω causing a continuous change in the dependent variables A , $\tilde{A}(\omega)$, & β , a visualization of these changes is visualized in Figure 2.1. This poses difficulty because the techniques used to produce A , $\tilde{A}(\omega)$, & β 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 X , Y , R & Ω into windows in time. The window, whose width will be represented by the variable $nspw[samples]$. There is a trade off between accuracy and resolution of variables as $nspw$ is changed. This trade off will be elaborated on in §??.

The windowing approach is visually depicted using Ω in figure 2.2 as it changes over time. When the width of the window is small enough, such as in Figure 2.2b, the change in the independent variable becomes vanishingly small. Other continuous variables such as A , $\tilde{A}(\omega)$, & β are expected to behave similarly, 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. Inside of a window of vibration, as the one shown in Figure 2.3 the Speed,



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

Figure 2.2: Rotor Spin Speed Windowing effect.

Amplitude and Phase all approach constant the smaller $nspw$ gets.

The variables $V(t)$, $W(t)$ & $R(t)$ will now take the form $V(n)$, $W(n)$ & $R(n)$ inside the window (fig. 2.3), where n is sample number. Spin Speed is simply taken as an average in the window, $\Omega(N) = \text{avg}(\Omega(0 : nspw))$, 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

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

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 $V(n)$ & $W(n)$ are presented in Figure 2.3 to aid in understanding the sections to follow.

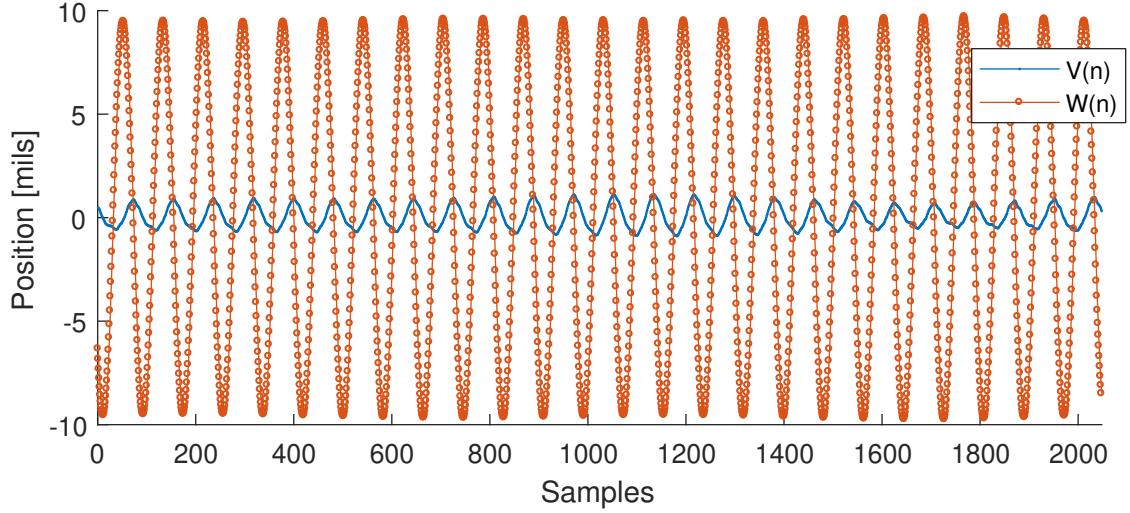


Figure 2.3: A window in time of the transient vibration signals for a window, $nspw$, of 2048[samples].

2.1.1 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.

2.1.2 Spectrum

The frequency spectrum of the signal is calculated inside the window using a Fourier Transform. In MATLAB the Fast Fourier Transform (fft) has been preprogrammed allowing easy working between time and frequency domains.

$$\tilde{A}_V = \frac{\text{fft}(V)}{nspw}, \quad \& \quad \tilde{A}_W = \frac{\text{fft}(W)}{nspw} \quad (2.1)$$

A useful way to represent the data is using complex variable to compact the two

orthogonal displacements V & W as

$$Z = V + iW \quad (2.2)$$

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

$$\tilde{A}_\pm = \frac{\text{fft}(Z)}{nspw} \quad (2.3)$$

Thus far, the frequency spectrum in the real coordinates and in the complex is in terms of samples on the dependent 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 $d\omega = f_s/nspw$. It is also known that the frequency vector is the same length as the time domain signal

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

and to center the spectrum at a frequency of 0

$$Q = \text{ceiling}((nspw + 1)/2)$$

$$\omega_Q = d\omega(Q - 1)$$

$$\omega = f - fQ$$

where ω , here in Hz , is the dependent variable that pairs with the real or complex Amplitude spectrum. The Amplitude spectrum in real coordinates(fig.(2.1)) is symmetric for positive and negative ω , so typically when the spectrum of a single variable is presented in a spectrum plot or in a cascade it is only the positive frequency side presented. On the other hand, the complex representation of the amplitude spectrum((2.3)) is not symmetric on the positive and negative sides of $/omega$. This decomposition can be better understood by realizing the form of the Fourier transform as the summation of circles in the complex plane, $\tilde{A}_\pm(\omega)e^{i\omega t}$. When ω is positive this represents a positive rotation, and when negative represents a negative rotation.

For a given whirl speed, ω_i , this results in sum of a positively rotating circle of amplitude, $\tilde{A}(\omega_i)$ and a negatively rotating circle of amplitude $\tilde{A}(-\omega_i)$. The ellipse formed by this summation is the orbit of the shaft centerline at this specific speed. With the understanding of contributions of $\tilde{A}(\omega)$ and $\tilde{A}(-\omega)$, we can realize that the resulting ellipse will rotate in the positive direction if $\tilde{A}(\omega) > \tilde{A}(-\omega)$ and in the negative direction otherwise.

2.1.3 Phase

2.1.3.1 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 using either a zero-crossing, peak-finding, or threshold algorithm can be employed to reference a specific angle of the shaft rotation. If the vibration signal is mostly synchronous, that is 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.

$$\beta_k = 2\pi \frac{\#ref_k - \#peak_k}{\#ref_k - \#ref_{k-1}} \quad (2.4)$$

where β_k is the phase lag of the signal of interest from the reference signal at the k th reference cycle, $\#ref_k$ is the sample number of the reference trigger, and $\#peak_k$ 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 current phase information on just the last rotation of the shaft. In the application to the window of vibration data, fig. 2.3, the measurements of each cycle would be averaged across the window as $\beta(\omega) = avg(\beta_k)$ for however many indexes k were found in the window.

2.1.3.2 Frequency Domain Approach

Alternatively, the phase angle can be determined using the frequency domain 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

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

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, $\omega = \Omega$.

2.2 Rotordynamic Figures

In the previous section Amplitude, Phase, and frequency spectrum were calculated for the interior of a window of index N . Each of the data 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 $NW = \frac{\text{length}(X)}{nspw}$ where X is a placeholder for any time domain signal. After all windows have been exhausted, vectors for A , $\tilde{A}(\omega)$, β & Ω will all be of length NW .

For the visualization of the plots, experimental data from a overhung rotor system with one disk will used to present the figures.

2.2.1 Bode

The Bode diagram for the example overhung rotor system is given in Figure 2.4. First, 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 separate transitions are evident. Having two separate peaks is an indication of high anisotropy of stiffness in the system. By observing the phase lag of each the vertical and the horizontal, it is evident that the orbit direction is opposite the spin speed between speeds 1280-1350[RPM]. This is due to the phase angles switching polarity with one another. If normally horizontal lags vertical, as is suggested by the sub-synchronous and super-synchronous range, then during the critical speed the orbit is reversed since vertical begins to lag horizontal.

Bode diagrams are extremely useful in diagnosing unbalance in the system through inspection of the phase lag portion. 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 through a natural frequency, and is at 90 deg during one.

2.2.2 Full Spectrum and Full Spectrum Cascade

A single Spectrum at a specific speed is shown as Figure 2.5. This is the Complex representation of the spectrum 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 negative whirl direction since the negative

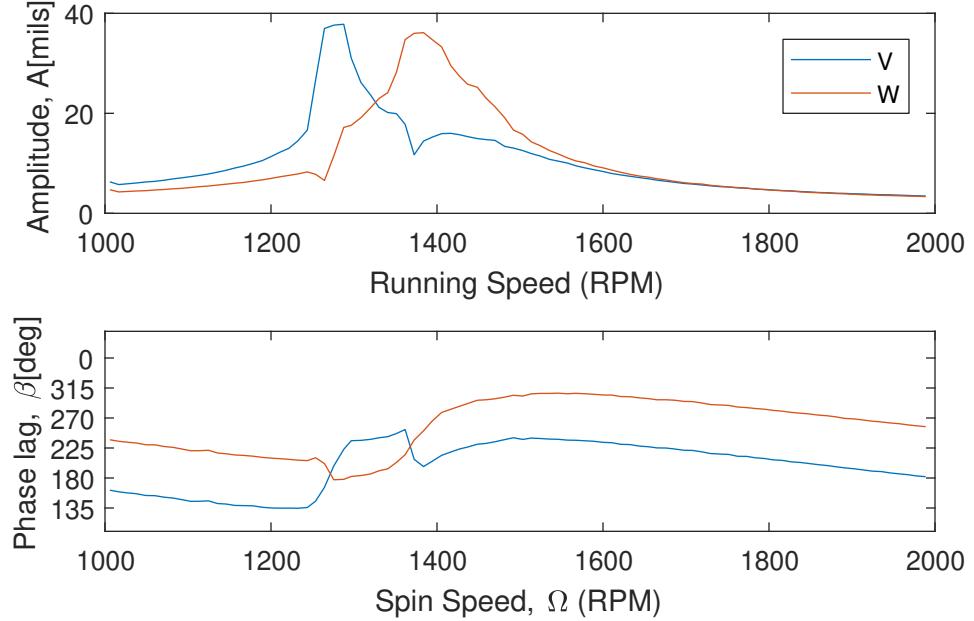


Figure 2.4: Bode diagram of the experimental example overhung system. Signals are filtered to synchronous speed.

amplitude, $\tilde{A}_{\pm}(-\omega) = 10.27$ at its peak is greater than $\tilde{A}_{\pm}(\omega) = 8.80$. Also, there is minimal amplitude in the spectrum other than this single frequency of 31.25[Hz] indicating the vibration is highly synchronous. A Cascade plot is demonstrated with the experimental system described in §2.2, as Figure 2.6. 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, $\tilde{A}_{\pm}(-\omega)$, leads to this phenomena.

The Cascade plot is particularly useful in characterizing non-synchronous vibration. Slightly evident in the example cascade of figure 2.6 is the super-synchronous vibration at twice the spin speed, this is often called the 2X vibration and likewise and other non-synchronous whirl can be referenced as nX. The cascade plot is an indispensable tool for the analysis of fluid film bearing as they are characterized by sub-synchronous whirl that is difficult to identify in a Bode Diagram.

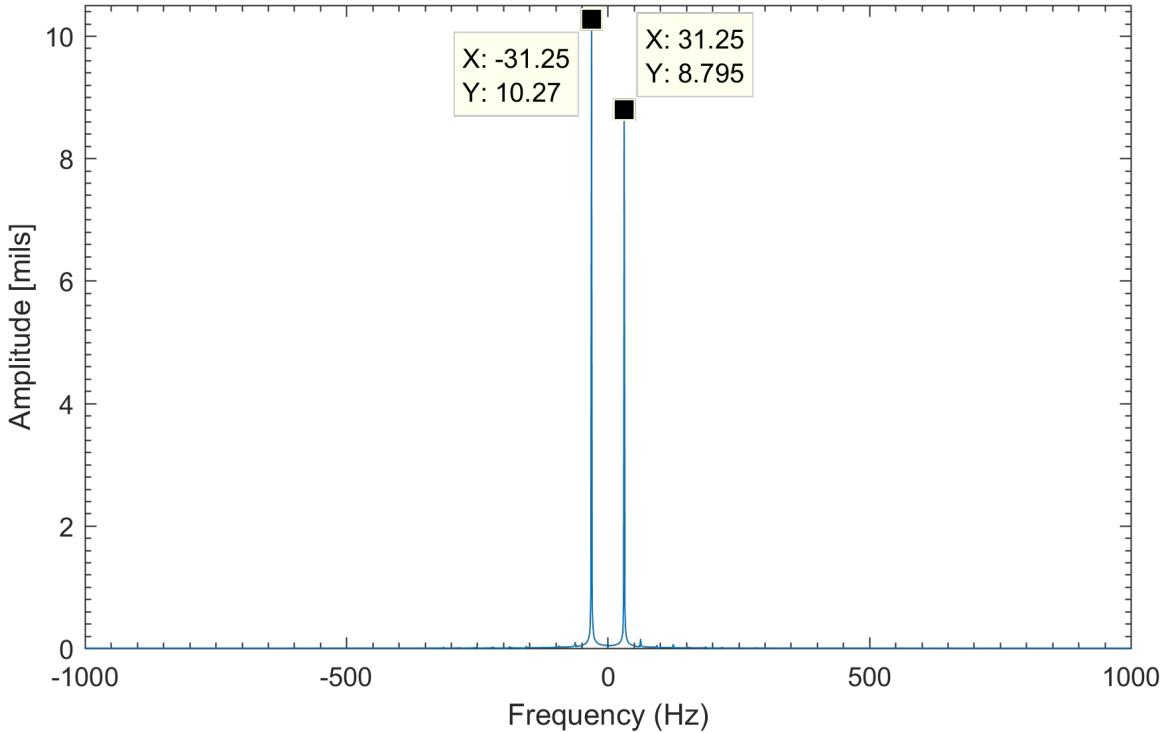


Figure 2.5: Example Full Spectrum.

2.2.3 Orbit

Now 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 3D Orbit allows for the visualization of complex phenomena in a simple intuitive way. Figure 2.7 3D Orbit is given for the experimental example described in §2.2. Appearing, once again is evidence of the negative whirl in the critical speed range. In the 3D orbit a necking of the shape can be seen between the speeds of 1200-1400[RPM] indicating the orbit has reversed its direction. Looking at independent orbits at specific speeds should explicitly demonstrate the orbit necking and turning negative. In Figure 2.8 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

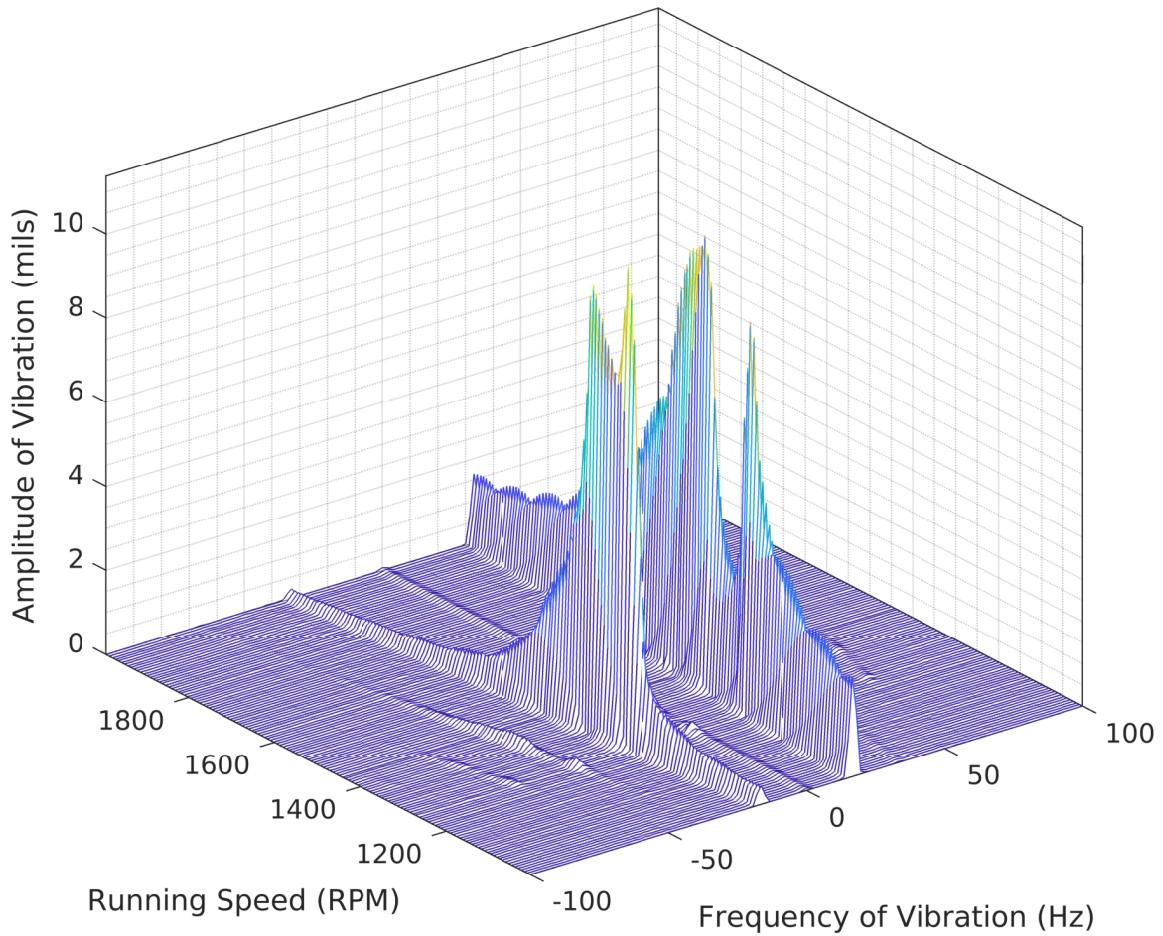


Figure 2.6: Cascade of the experimental system described in §2.2.

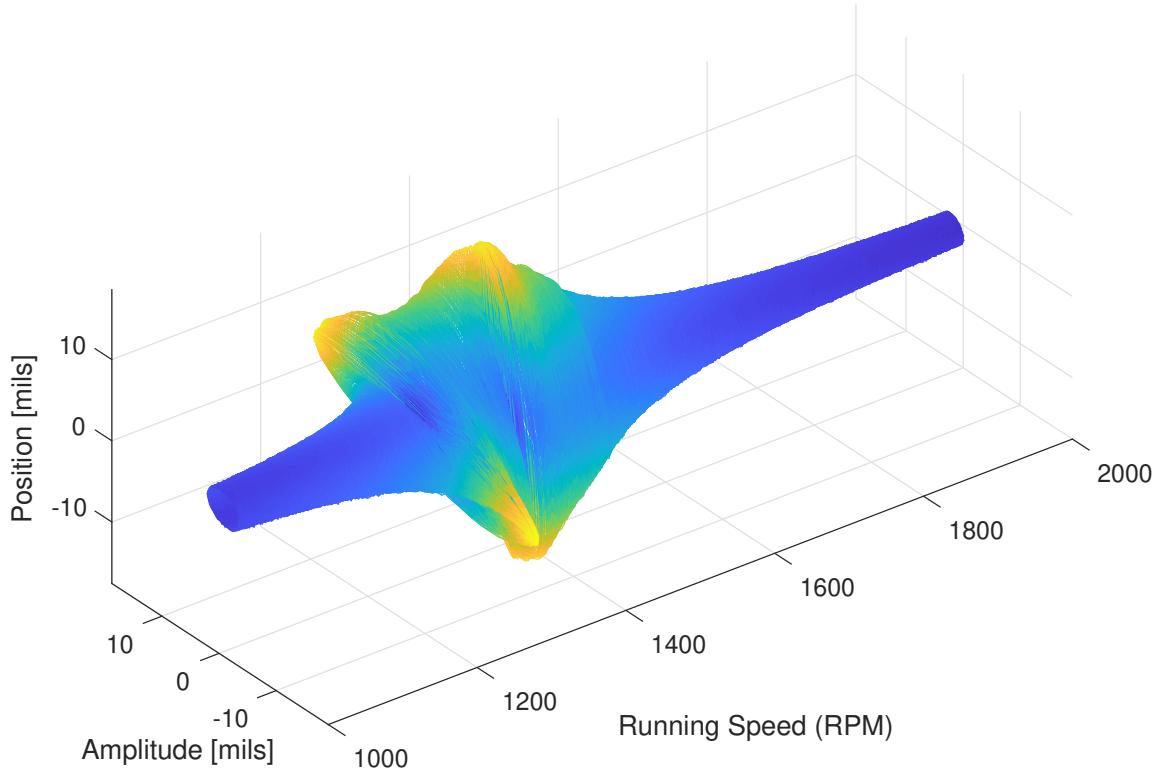


Figure 2.7: 3D Orbit of the experimental system described in §2.2. Lighter colors indicate larger vibration.

process is reversed by speed 1289[RPM]. Therefore, it can be confirmed that the orbit is whirling backward between the speeds 1205- 1280[RPM].

2.2.4 Filtering

Correlating phase angles between real signals can be extremely difficult due to the noise and harmonic frequencies that may disrupt the measurement. 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.

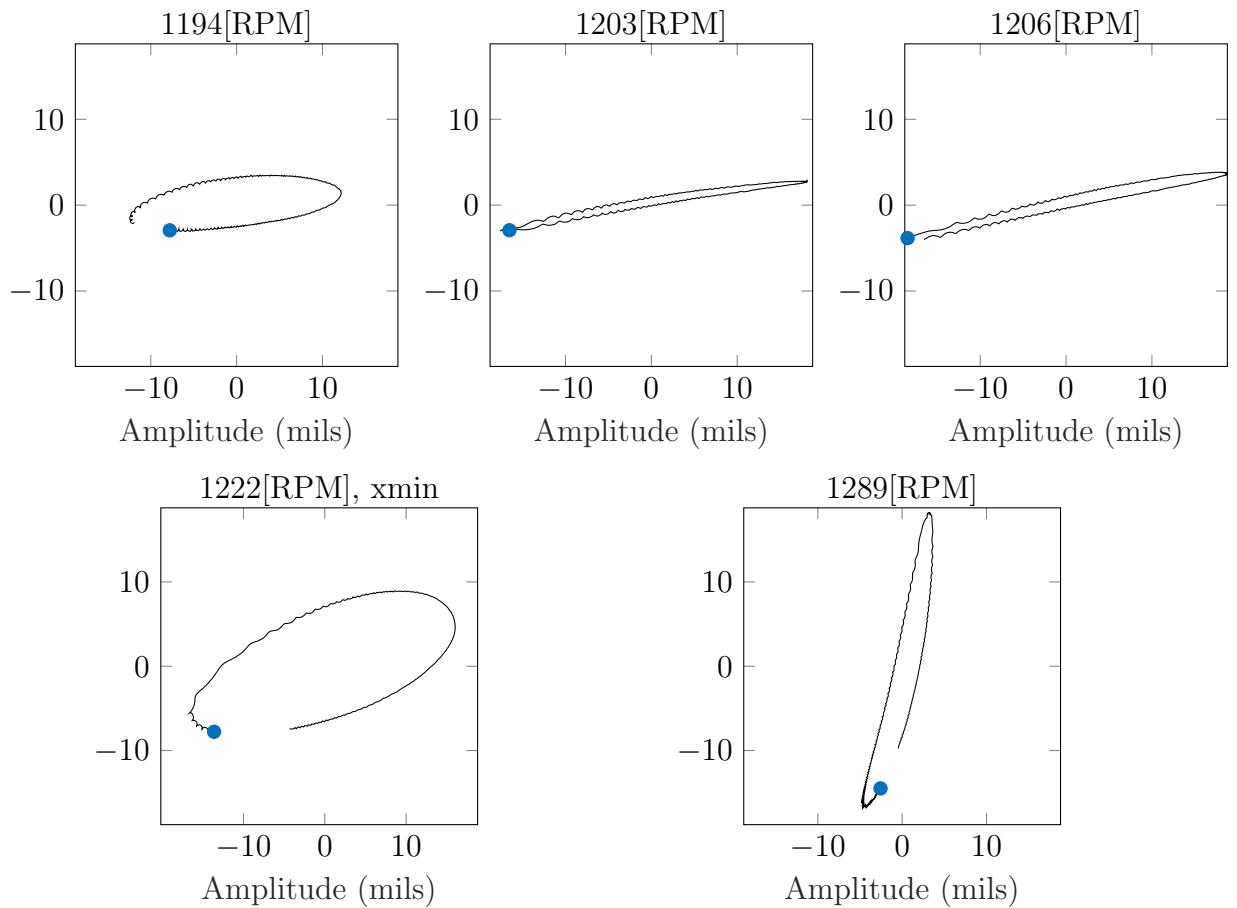


Figure 2.8: Orbita de l'exemple experimental. La velocitat d'escrivella és contrària al sentit dels agujers. Els punts blaus indiquen la posició de referència del tija, i el començament de cada òbita.

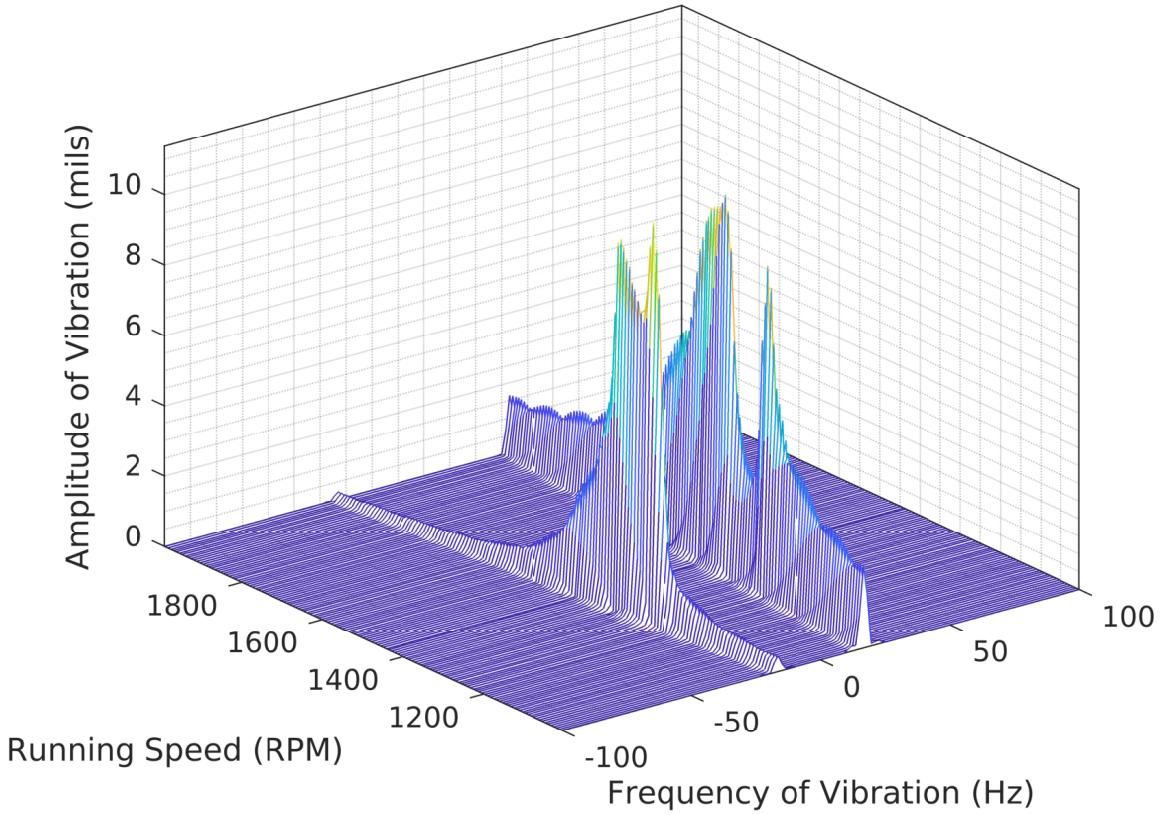


Figure 2.9: Cascade of the experimental system with a synchronous filter applied.

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 lose dynamic information from one step to the next.

A synchronous filter was applied to the experimental example and the cascade plot is shown in Figure 2.9. 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 Bode plot. But, with many real systems filtering will be necessary to analyze the system.

Chapter 3

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 discretizing method to solve problems of rotating machines. One of which is the generic form that the equation, and its parts take. The method lends itself to being able to easily create new matrices that can insert right into your equations of motion. Another, is the ability to move components around in space without the need to reevaluate the physics. Finally, the analysis techniques for Multiple Degree Of Freedom (MDOF) systems is wide reaching, and will serve to richly enhance our understanding of a complex system made of simple parts.

First, the beam element commonly referred to as the Timoshenko Beam Element will be derived from the kinematic and constitutive constraints (An alternative element, the Bernoulli-Euler element is provided in Appendix A). Then 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. Further, an extension to the beam element model is presented that will consider the viscous damping effects in the beam element. Equations for disks, bearings, and complex versions of all equations will be presented. Finally, the assembly and analysis of the model will result in intuitive figures that will be shown to compare well to experimental data.

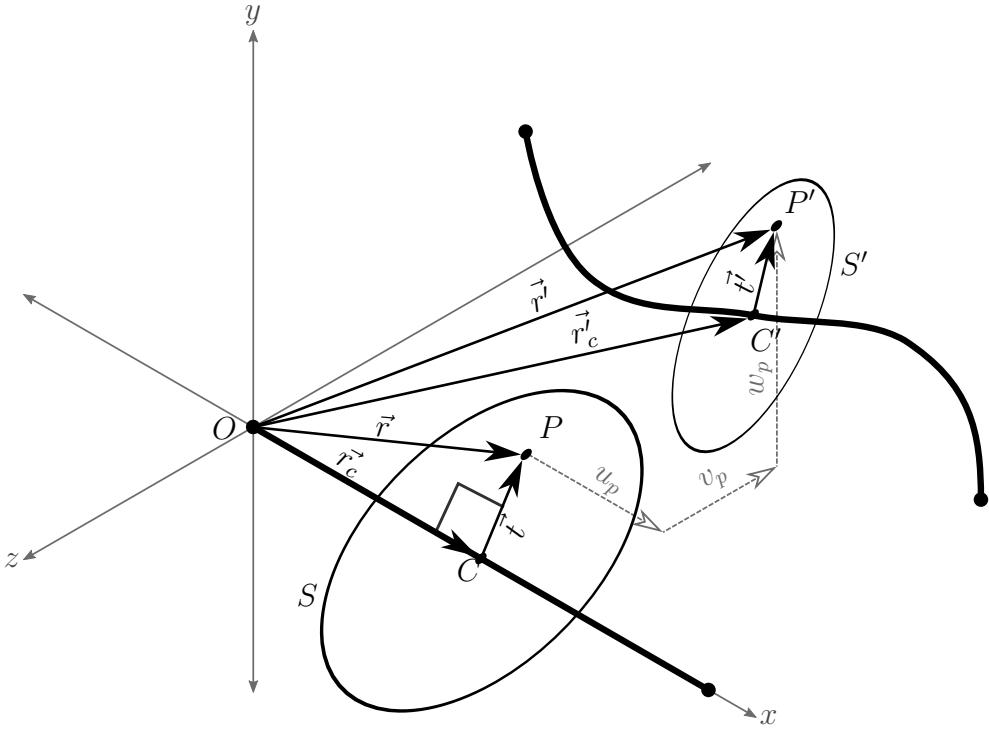


Figure 3.1: Beam Element with nodal displacements.

3.1 Beam Element FE Equation

3.1.1 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 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. So then all displacements are functions of time, t , and the axial spacial coordinate, x .

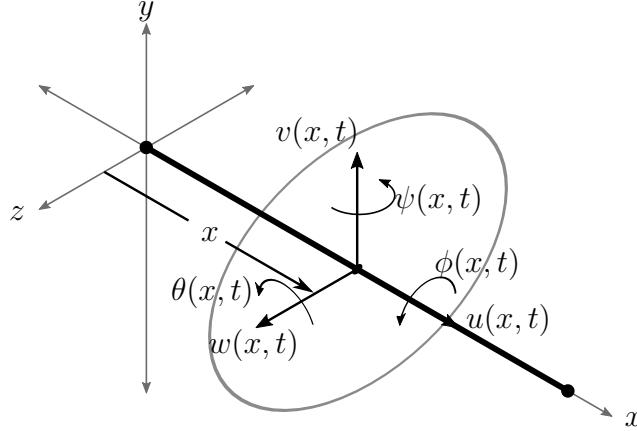


Figure 3.2: Timoshenko beam section with degrees of freedom at some point x along beam axis.

3.1.1.1 Kinematic Relationships

In order to develop the stresses and strains internal to 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 3.1. 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 vector $\vec{t} = \overrightarrow{CP}$ points from the beam 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, ψ, θ , & 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

$$\vec{r} = \vec{r}_c + \vec{t} \quad (3.1)$$

Rotations are represented with a rotation transformation matrix,

$$\vec{t}' = \underline{R} \vec{t} \quad (3.2)$$

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

$$\underline{R} = \begin{bmatrix} 1 & -\theta & \psi \\ \theta & 1 & -\phi \\ -\psi & \phi & 1 \end{bmatrix} \quad (3.3)$$

and the translation with a displacement vector

$$\vec{r}'_c = \vec{r}_c + \vec{u} \quad (3.4)$$

Combined motion from P to P' can be defined by

$$\vec{u}_p = \vec{r}' - \vec{r} \quad (3.5)$$

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

$$\vec{u}_p = \vec{r}'_c + \vec{t}' - \vec{r}_c + \vec{t} \quad (3.6)$$

and now using the definition for \vec{u} and \vec{t} leads to the simplified expression for the motion of P

$$\vec{u}_p = \vec{u} + (\underline{R} - \underline{I})\vec{t} \quad (3.7)$$

expanding matrices reveals the equation

$$\vec{u}_p = \begin{Bmatrix} u_p \\ v_p \\ w_p \end{Bmatrix} = \begin{Bmatrix} u \\ v \\ w \end{Bmatrix} + \begin{bmatrix} 0 & -\theta & \psi \\ \theta & 0 & -\phi \\ -\psi & \phi & 0 \end{bmatrix} \begin{Bmatrix} 0 \\ y \\ z \end{Bmatrix} \quad (3.8)$$

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

$$\vec{u}_p = \begin{Bmatrix} u - \theta y + \psi z \\ v - \phi z \\ w + \phi y \end{Bmatrix} \quad (3.9)$$

3.1.1.2 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

$$\sigma_{ij} = \begin{bmatrix} \sigma_{xx} & \sigma_{xy} & \sigma_{xz} \\ \sigma_{xy} & 0 & 0 \\ \sigma_{xz} & 0 & 0 \end{bmatrix} \quad (3.10)$$

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

$$\epsilon_{ij} = \frac{1}{E}[(1 + \nu)\sigma_{ij} - \nu\delta_{ij}\sigma_{kk}] \quad (3.11)$$

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

$$\begin{Bmatrix} \sigma_{xx} \\ \sigma_{xy} \\ \sigma_{xz} \end{Bmatrix} = \begin{bmatrix} E & 0 & 0 \\ 0 & 2G & 0 \\ 0 & 0 & 2G \end{bmatrix} \begin{Bmatrix} \epsilon_{xx} \\ \epsilon_{xy} \\ \epsilon_{xz} \end{Bmatrix} \quad (3.12)$$

where, $G = \frac{E}{2(1+\nu)}$. Strains are derived from displacements of equation (3.9) using a linear strain-displacement relationship for infinitesimal strains: $2\epsilon_{ij} = u_{i,j} + u_{j,i}$.

$$\begin{cases} \epsilon_{xx} = u' - \theta'y + \psi'z \\ \epsilon_{xy} = \frac{1}{2}(v' - \phi'z - \theta) \\ \epsilon_{xz} = \frac{1}{2}(w' + \phi'y + \psi) \end{cases} \quad (3.13)$$

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.

$$\begin{cases} \varepsilon = u' \\ \rho_y = -\theta' \\ \rho_z = \psi' \\ \varphi = \phi' \\ \gamma_y = v' - \theta \\ \gamma_z = w' + \psi \end{cases} \quad (3.14)$$

allowing the representation of the strains from eq (3.13) using the generalized strains as:

$$\begin{cases} \epsilon_{xx} = \varepsilon + \rho_y y + \rho_z z \\ \epsilon_{xy} = \frac{1}{2}(\gamma_y - \varphi z) \\ \epsilon_{xz} = \frac{1}{2}(\gamma_z + \varphi y) \end{cases} \quad (3.15)$$

Now the stresses in equation (3.12) can be represented with the displacements as

$$\begin{Bmatrix} \sigma_{xx} \\ \sigma_{xy} \\ \sigma_{xz} \end{Bmatrix} = \begin{Bmatrix} E(u' - \theta'y + \psi'z) \\ G(v' - \phi'z - \theta) \\ G(w' + \phi'y + \psi) \end{Bmatrix} = \begin{Bmatrix} E(\varepsilon + \rho_y y + \rho_z z) \\ G(\gamma_y - \varphi z) \\ G(\gamma_z + \varphi y) \end{Bmatrix} \quad (3.16)$$

$$U = \int_V \sigma_{ij} \epsilon_{ij} dV \quad (3.17)$$

Now the inner product of the total virtual strain energy can be expanded

$$U = \int_V [\sigma_{xx} \epsilon_{xx} + 2\sigma_{xy} \epsilon_{xy} + 2\sigma_{xz} \epsilon_{xz}] dV \quad (3.18)$$

Then expand virtual strains into the generalized strains corresponding to the degrees of freedom of the beam element and collect terms on generalized strains

$$U = \int_V [\sigma_{xx} \varepsilon + \sigma_{xx} y \rho_y + \sigma_{xx} z \rho_z + \sigma_{xy} \gamma_y + \sigma_{xz} \gamma_z + (\sigma_{xz} y - \sigma_{xy} z) \varphi] dV \quad (3.19)$$

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

$$\left\{ \begin{array}{lll} N = \int_A \sigma_{xx} dA & = E(A\varepsilon + S_y\rho_y + S_z\rho_z) \\ M_y = \int_A \sigma_{xx} z dA & = E(S_z\varepsilon + I_{xy}\rho_y + I_y\rho_z) \\ M_z = \int_A \sigma_{xx} y dA & = E(S_y\varepsilon + I_z\rho_y + I_{xy}\rho_z) \\ Q_y = \int_A \sigma_{xy} dA & = \kappa G(A\gamma_y - S_z\varphi) \\ Q_z = \int_A \sigma_{xz} dA & = \kappa G(A\gamma_z + S_y\varphi) \\ M_x = \int_A (\sigma_{xz}y - \sigma_{xy}z) dA & = \kappa G(A_y\gamma_z - A_z\gamma_y + J_x\varphi) \end{array} \right. \quad (3.20)$$

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

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

$$\left\{ \begin{array}{ll} N = EA\varepsilon & = EAu' \\ M_y = EI_y\rho_z & = EI_y\psi' \\ M_z = EI_z\rho_y & = -EI_z\theta' \\ Q_y = \kappa GA\gamma_y & = \kappa GA(v' - \theta) \\ Q_z = \kappa GA\gamma_z & = \kappa GA(w' + \psi) \\ M_x = \kappa GJ_x\varphi & = \kappa GJ_x\phi' \end{array} \right. \quad (3.21)$$

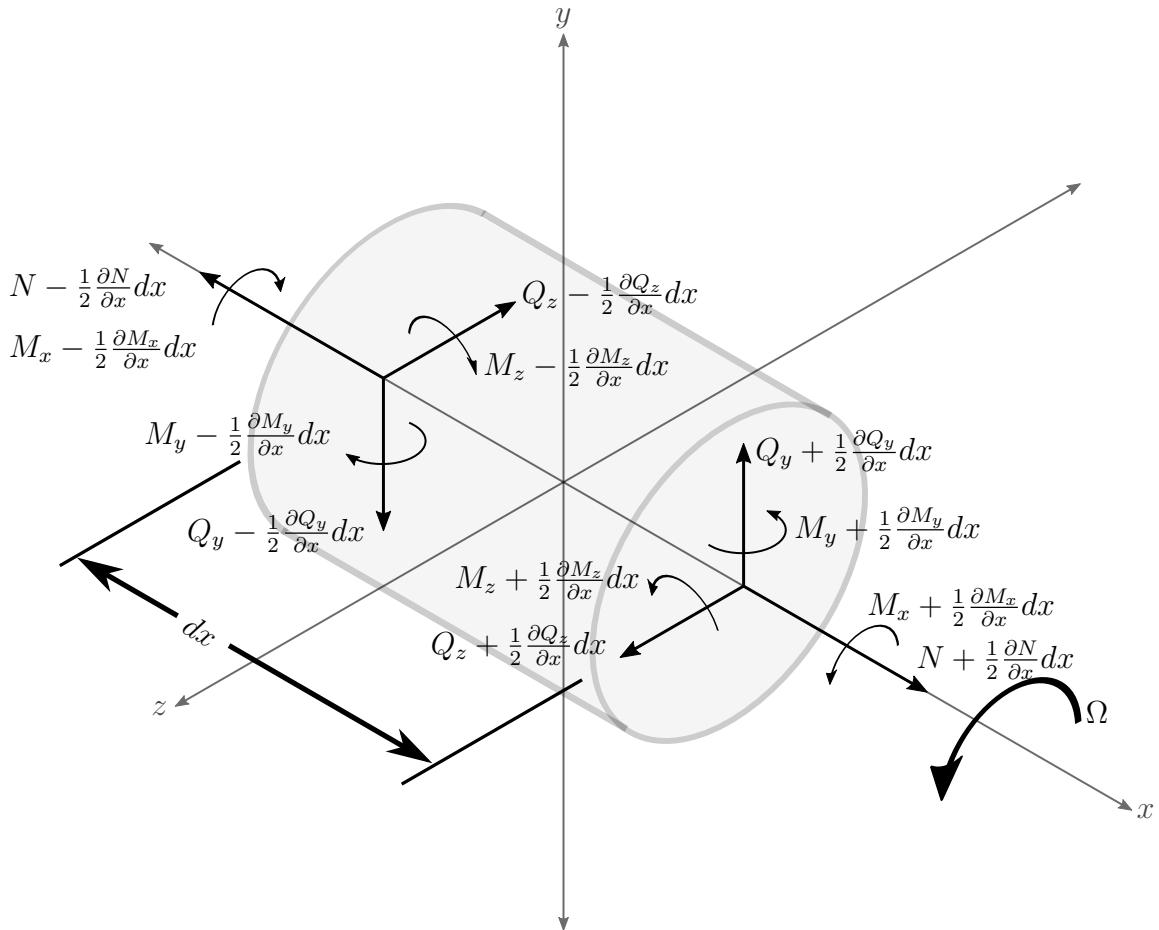


Figure 3.3: Beam differential element with generalized forces.

3.1.1.3 Differential Equations of Motion

Now the equations of motion are 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 3.3 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. Applying summation of forces in the x-direction

$$(N + \frac{1}{2} \frac{\partial N}{\partial x} dx) - (N - \frac{1}{2} \frac{\partial N}{\partial x} dx) = \rho A dx \frac{\partial^2 u}{\partial x^2} \quad (3.22)$$

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

$$\left\{ \begin{array}{l} \frac{\partial N}{\partial x} = \rho A \frac{\partial^2 u}{\partial x^2} \\ \frac{\partial Q_y}{\partial x} = \rho A \frac{\partial^2 v}{\partial x^2} \\ \frac{\partial Q_z}{\partial x} = \rho A \frac{\partial^2 w}{\partial x^2} \\ \frac{\partial M_y}{\partial x} - Q_z = \rho I_y \frac{\partial^2 \psi}{\partial x^2} + \rho J_x \Omega \frac{\partial \theta}{\partial x} \\ \frac{\partial M_z}{\partial x} + Q_y = \rho I_z \frac{\partial^2 \theta}{\partial x^2} - \rho J_x \Omega \frac{\partial \psi}{\partial x} \\ \frac{\partial M_x}{\partial x} = \rho J_x \frac{\partial^2 \phi}{\partial x^2} \end{array} \right. \quad (3.23)$$

Gyroscopic moments have been explicitly added to the summations in the appropriate equations. The generalized forces of equation (3.21) are substituted in the equilibrium equations (3.23)

$$\left\{ \begin{array}{l} EAu'' = \rho A \ddot{u} \\ \kappa GA(v'' - \theta') = \rho A \ddot{v} \\ \kappa GA(w'' + \psi') = \rho A \ddot{w} \\ EI_y \psi'' - \kappa GA(w' + \psi) = \rho I_y \ddot{\psi} + \rho J_x \Omega \dot{\theta} \\ EI_z \theta'' + \kappa GA(v' - \theta) = \rho I_z \ddot{\theta} - \rho J_x \Omega \dot{\psi} \\ \kappa G J_x \phi'' = \rho J_x \ddot{\phi} \end{array} \right. \quad \begin{array}{l} (3.24a) \\ (3.24b) \\ (3.24c) \\ (3.24d) \\ (3.24e) \\ (3.24f) \end{array}$$

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

$$\underline{\mathcal{M}}^e \ddot{\vec{u}} + \Omega \underline{\mathcal{G}}^e \dot{\vec{u}} - \left(\frac{\partial(\cdot)}{\partial x} \underline{\mathcal{S}}^e - \underline{\mathcal{P}}^e \underline{\mathcal{S}}^e \right) \vec{u} = 0 \quad (3.25)$$

where,

$$\left\{ \begin{array}{l} \underline{\mathcal{M}}^e = \begin{bmatrix} \rho A & 0 & 0 & 0 & 0 & 0 \\ 0 & \rho A & 0 & 0 & 0 & 0 \\ 0 & 0 & \rho I_y & 0 & 0 & 0 \\ 0 & 0 & 0 & \rho I_z & 0 & 0 \\ 0 & 0 & 0 & 0 & \rho A & 0 \\ 0 & 0 & 0 & 0 & 0 & \rho J_x \end{bmatrix} \quad \underline{\mathcal{G}}^e = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \rho J_x & 0 & 0 \\ 0 & 0 & -\rho J_x & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \\ \\ \underline{\mathcal{P}}^e = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad \underline{\mathcal{S}}^e = \begin{bmatrix} \kappa G A \frac{\partial()}{\partial x} & 0 & 0 & -\kappa G A & 0 & 0 \\ 0 & \kappa G A \frac{\partial()}{\partial x} & \kappa G A & 0 & 0 & 0 \\ 0 & 0 & E I_y \frac{\partial()}{\partial x} & 0 & 0 & 0 \\ 0 & 0 & 0 & E I_z \frac{\partial()}{\partial x} & 0 & 0 \\ 0 & 0 & 0 & 0 & E A \frac{\partial()}{\partial x} & 0 \\ 0 & 0 & 0 & 0 & 0 & \kappa G J_x \frac{\partial()}{\partial x} \end{bmatrix} \end{array} \right. \quad (3.26)$$

and $\vec{\mathbf{u}} = [v, w, \psi, \theta, u, \phi]^T$. 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.

$$\int_0^l \delta \vec{\mathbf{u}}^T \underline{\mathcal{M}}^e \ddot{\vec{\mathbf{u}}} dx + \int_0^l \delta \vec{\mathbf{u}}^T \Omega \underline{\mathcal{G}}^e \dot{\vec{\mathbf{u}}} - \int_0^l \delta \vec{\mathbf{u}}^T \frac{\partial()}{\partial x} \underline{\mathcal{S}}^e \vec{\mathbf{u}} dx + \int_0^l \delta \vec{\mathbf{u}}^T \underline{\mathcal{P}} \underline{\mathcal{S}}^e \vec{\mathbf{u}} dx = 0 \quad (3.27)$$

integration by parts on the third term and replacing $\underline{\mathcal{S}}$ with $\underline{\mathcal{D}} \underline{\mathcal{B}}$, 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

$$\int_0^l \delta \vec{\mathbf{u}}^T \underline{\mathcal{M}}^e \ddot{\vec{\mathbf{u}}} dx + \Omega \int_0^l \delta \vec{\mathbf{u}}^T \underline{\mathcal{G}}^e \dot{\vec{\mathbf{u}}} + \int_0^l \delta \vec{\mathbf{u}}^T \left(\frac{\partial()}{\partial x} \underline{\mathbf{I}} + \underline{\mathcal{P}} \right) \underline{\mathcal{D}}^e \underline{\mathcal{B}} \vec{\mathbf{u}} dx = 0 \quad (3.28)$$

$$\underline{\mathcal{D}}^e = \begin{bmatrix} \kappa G A & 0 & 0 & 0 & 0 & 0 \\ 0 & \kappa G A & 0 & 0 & 0 & 0 \\ 0 & 0 & E I_y & 0 & 0 & 0 \\ 0 & 0 & 0 & E I_z & 0 & 0 \\ 0 & 0 & 0 & 0 & E A & 0 \\ 0 & 0 & 0 & 0 & 0 & \kappa G J_x \end{bmatrix} \quad \& \quad \underline{\mathcal{B}} = \begin{bmatrix} \frac{\partial()}{\partial x} & 0 & 0 & -1 & 0 & 0 \\ 0 & \frac{\partial()}{\partial x} & 1 & 0 & 0 & 0 \\ 0 & 0 & \frac{\partial()}{\partial x} & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{\partial()}{\partial x} & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{\partial()}{\partial x} & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{\partial()}{\partial x} \end{bmatrix} \quad (3.29)$$

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

$$\int_0^l \delta \vec{\mathbf{u}}^T \underline{\mathcal{M}}^e \ddot{\vec{\mathbf{u}}} dx + \Omega \int_0^l \delta \vec{\mathbf{u}}^T \underline{\mathcal{G}}^e \dot{\vec{\mathbf{u}}} + \int_0^l \delta \vec{\mathbf{u}}^T \underline{\mathcal{B}}^T \underline{\mathcal{D}}^e \underline{\mathcal{B}} \vec{\mathbf{u}} dx = 0 \quad (3.30)$$

$\underline{\mathcal{M}}^e$ 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.

3.1.1.4 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.

$$\begin{cases} \vec{\mathbf{u}}(x, t) = \underline{\mathbf{N}}(x)\vec{\mathbf{q}}(t) \\ \dot{\vec{\mathbf{u}}}(x, t) = \underline{\mathbf{N}}(x)\dot{\vec{\mathbf{q}}}(t) \\ \ddot{\vec{\mathbf{u}}}(x, t) = \underline{\mathbf{N}}(x)\ddot{\vec{\mathbf{q}}}(t) \\ \delta\vec{\mathbf{u}}(x, t) = \underline{\mathbf{N}}(x)\delta\vec{\mathbf{q}}(t) \end{cases} \quad (3.31)$$

where, $\vec{\mathbf{u}} = [v, w, -\psi, \theta, u, \phi]^\top$ & $\vec{\mathbf{q}} = [v_1, w_1, -\psi_1, \theta_1, v_2, w_2, -\psi_2, \theta_2, u_1, \phi_1, u_2, \phi_2]^\top$. This specific 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 solve the static portion of the differential equations (3.24). These shape functions are chosen as polynomials that satisfy the boundary nodal displacements and rotations at the ends of a beam element. These nodal degrees of freedom, depicted in Figure 3.4 are considered to be interpolated through the beam element by the shape functions. Interpolation functions chosen are listed in Equation (3.32). Axial displacement, u , and torsional rotation, ϕ are independent, so their shape functions are

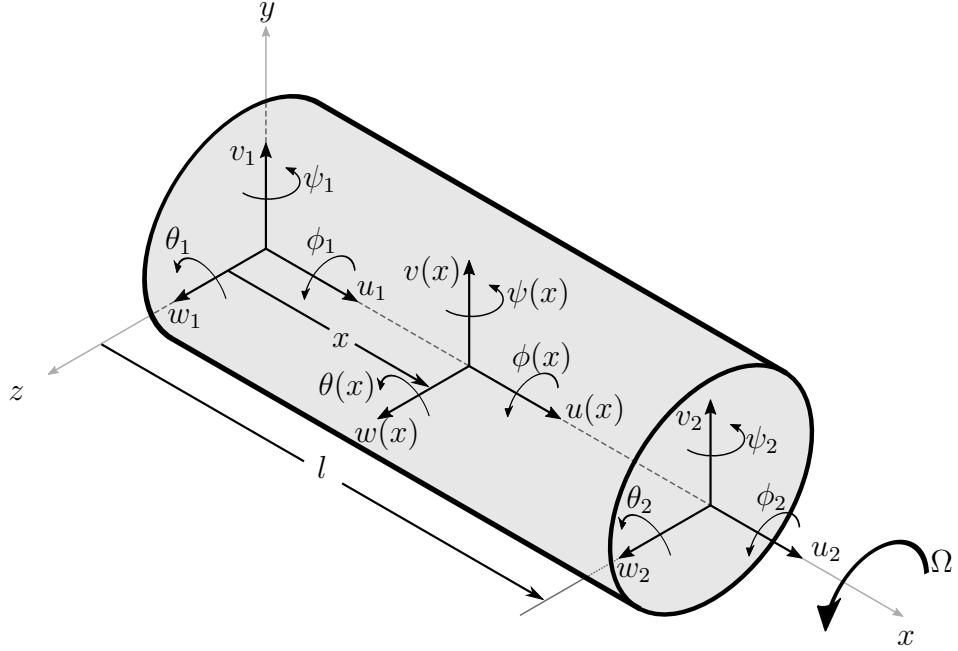


Figure 3.4: Beam Element with nodal displacements.

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 [26]. Polynomial functions are chosen for v & w and their rotational counterparts are derived using the differential relations.

$$\left\{ \begin{array}{l} u = c_1 + c_2 x \\ v = c_3 + c_4 x + c_5 x^2 + c_6 x^3 \\ w = c_7 + c_8 x + c_9 x^2 + c_{10} x^3 \\ \phi = c_{11} + c_{12} x \end{array} \right. \quad (3.32a)$$

$$(3.32b)$$

$$(3.32c)$$

$$(3.32d)$$

$c_{1,2,\dots}$ are the unknown constants of the polynomial solutions. Using transverse displacement of equations (3.32b) & (3.32c) in the differential equations (3.24b), (3.24c),

(3.24d), (3.24e) the interpolation functions of bending rotations are derived as:

$$\left\{ \begin{array}{l} \psi = K_y c_{10} - c_8 - 2c_9x - 3c_{10}x^2 \\ \theta = K_z c_6 + c_4 + 2c_5x + 3c_6x^2 \end{array} \right. \quad (3.33a)$$

$$\left\{ \begin{array}{l} \psi = K_y c_{10} - c_8 - 2c_9x - 3c_{10}x^2 \\ \theta = K_z c_6 + c_4 + 2c_5x + 3c_6x^2 \end{array} \right. \quad (3.33b)$$

where, $K_y = \frac{6EI_y}{\kappa GA}$ & $K_z = \frac{6EI_z}{\kappa GA}$ Boundary Conditions Boundary conditions for the interpolation polynomials of equations (3.32) & (3.33) are defined as the components of the vector $\vec{\mathbf{q}} u_j = u(x_j)$ and similarly 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 this relation between the polynomial constants and the boundary conditions.

$$\left\{ \begin{array}{l} u_1 \\ u_2 \\ v_1 \\ v_2 \\ w_1 \\ w_2 \\ \psi_1 \\ \psi_2 \\ \theta_1 \\ \theta_2 \\ \phi_1 \\ \phi_2 \end{array} \right\} = \left[\begin{array}{cccccccccccc} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & l & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & l & l^2 & l^3 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & l & l^2 & l^3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & -K_y & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & -2l & -K_y - 3l^2 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & K_z & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 2l & K_z + 3l^2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & l \end{array} \right] \left\{ \begin{array}{l} c_1 \\ c_2 \\ c_3 \\ c_4 \\ c_5 \\ c_6 \\ c_7 \\ c_8 \\ c_9 \\ c_{10} \\ c_{11} \\ c_{12} \end{array} \right\} \quad (3.34)$$

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

(3.32) & (3.33) giving the interpolations as functions of the nodal displacements.

$$\begin{cases} u = N_1 u_1 + N_2 u_2 \\ v = T_{t_1y} v_1 + T_{t_2y} v_2 + T_{r_1y} \theta_1 + T_{r_2y} \theta_2 \\ w = T_{t_1z} w_1 + T_{t_2z} w_2 + T_{r_1z} \psi_1 + T_{r_2z} \psi_2 \\ \psi = R_{t_1z} w_1 + R_{t_2z} w_2 + R_{r_1z} \psi_1 + R_{r_2z} \psi_2 \\ \theta = R_{t_1y} v_1 + R_{t_2y} v_2 + R_{r_1y} \theta_1 + R_{r_2y} \theta_2 \\ \phi = N_1 \phi_1 + N_2 \phi_2 \end{cases} \quad (3.35)$$

with;

$$\begin{cases} N_1 = 1 - \zeta & N_2 = \zeta \\ T_{t_1y,z} = \frac{1}{1+\alpha_{y,z}}(2\zeta^3 - 3\zeta^2 - \alpha_{y,z}\zeta + 1 + \alpha_{y,z}) & T_{t_2y,z} = \frac{1}{1+\alpha_{y,z}}(-2\zeta^3 + 3\zeta^2 + \alpha_{y,z}\zeta) \\ T_{r_1y,z} = \frac{l}{1+\alpha_{y,z}}[\zeta^3 - (2 + \frac{1}{2}\alpha_{y,z})\zeta^2 + (1 + \frac{1}{2}\alpha_{y,z})\zeta] & T_{r_2y,z} = \frac{l}{1+\alpha_{y,z}}[\zeta^3 - (1 - \frac{1}{2}\alpha_{y,z})\zeta^2 - \frac{1}{2}\alpha_{y,z}\zeta] \\ R_{t_1y,z} = \frac{6/l}{1+\alpha_{y,z}}(\zeta^2 - \zeta) & R_{t_2y,z} = \frac{6/l}{1+\alpha_{y,z}}(-\zeta^2 + \zeta) \\ R_{r_1y,z} = \frac{1}{1+\alpha_{y,z}}(3\zeta^2 - (4 + \alpha_{y,z})\zeta + 1 + \alpha_{y,z}) & R_{r_2y,z} = \frac{1}{1+\alpha_{y,z}}(3\zeta^2 - (2 - \alpha_{y,z})\zeta) \end{cases} \quad (3.36)$$

where, $\alpha_y = 2K_y/l^2 = \frac{12EI_y}{\kappa Gal^2}$, $\alpha_z = 2K_z/l^2 = \frac{12EI_z}{\kappa Gal^2}$, & $\zeta = x/l$. (3.35) is expressed in matrix form as it appears in (3.31) where

$$\mathbf{N}(x) = \begin{bmatrix} T_{t_1y} & 0 & 0 & T_{r_1y} & T_{t_2y} & 0 & 0 & T_{r_2y} & 0 & 0 & 0 & 0 \\ 0 & T_{t_1z} & T_{r_1z} & 0 & 0 & T_{t_2z} & T_{r_2z} & 0 & 0 & 0 & 0 & 0 \\ 0 & R_{t_1z} & R_{r_1z} & 0 & 0 & R_{t_2z} & R_{r_2z} & 0 & 0 & 0 & 0 & 0 \\ R_{t_1y} & 0 & 0 & R_{r_1y} & R_{t_2y} & 0 & 0 & R_{r_2y} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & N_1 & 0 & N_2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & N_1 & 0 & N_2 & 0 \end{bmatrix} \quad (3.37)$$

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

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 (3.36) is given in figure 3.5. Shape functions plotted with respect to the non-dimensional length lend a visu-

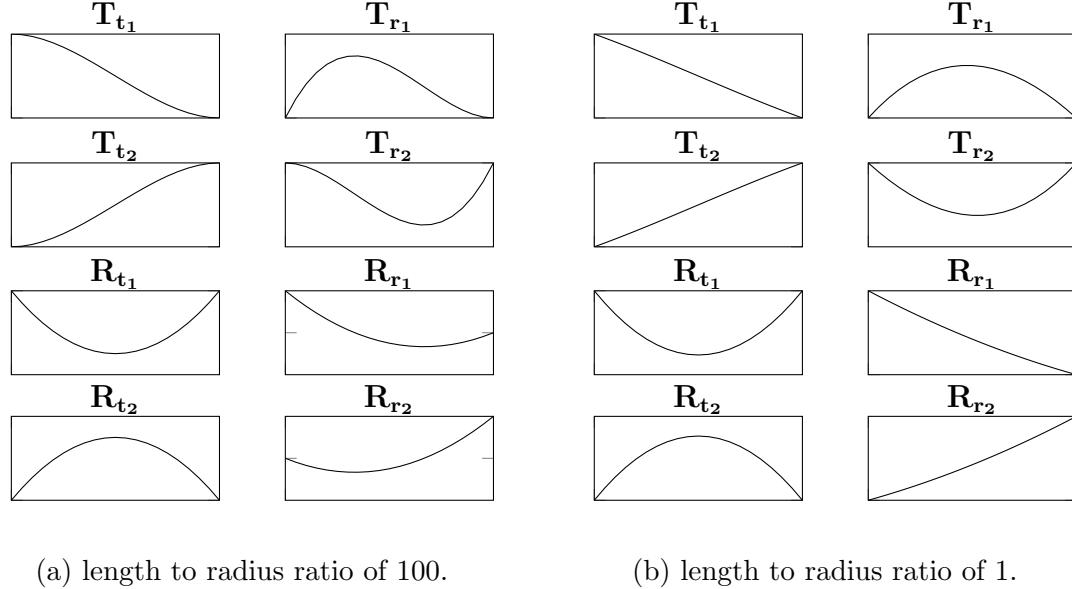


Figure 3.5: Shape Functions as they vary with ζ using two different ratios of length to radius of beam element.

alization 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 input, being the coordinate multiplied to it, to the output of the variable function. 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, the expected shape breaks as the radius approaches the length as in Figure 3.5b. All shapes this beam will make in the model is a linear combination of the shapes shown here.

3.1.1.5 Finite Equations of Motion

To obtain the equations of motion in terms of the generalized coordinates, \vec{q} , displacement variables \vec{u} are replaced with definitions in equation 3.31.

$$\int_0^l \underline{\mathbf{N}}^\top \delta \vec{\mathbf{q}}^\top \underline{\mathcal{M}}^e \underline{\mathbf{N}} \ddot{\vec{\mathbf{q}}} dx + \Omega \int_0^l \underline{\mathbf{N}}^\top \delta \vec{\mathbf{q}}^\top \underline{\mathcal{G}}^e \underline{\mathbf{N}} \dot{\vec{\mathbf{q}}} dx + \int_0^l \underline{\mathbf{N}}^\top \delta \vec{\mathbf{q}}^\top \underline{\mathcal{B}}^\top \underline{\mathcal{D}}^e \underline{\mathcal{B}} \underline{\mathbf{N}} \vec{\mathbf{q}} dx = 0 \quad (3.38)$$

Note that $\vec{\mathbf{q}}$ is not dependent on x so it, and its 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

$$\int_0^l \underline{\mathbf{N}}^\top \underline{\mathcal{M}}^e \underline{\mathbf{N}} dx \ddot{\vec{\mathbf{q}}} + \Omega \int_0^l \underline{\mathbf{N}}^\top \underline{\mathcal{G}}^e \underline{\mathbf{N}} dx \dot{\vec{\mathbf{q}}} + \int_0^l \underline{\mathbf{B}}^\top \underline{\mathcal{D}}^e \underline{\mathbf{B}} dx \vec{\mathbf{q}} = 0 \quad (3.39)$$

Define

$$\left\{ \begin{array}{l} \underline{\mathbf{M}}^e = \int_0^l \underline{\mathbf{N}}^\top \underline{\mathcal{M}}^e \underline{\mathbf{N}} dx \\ \underline{\mathbf{G}}^e = \int_0^l \underline{\mathbf{N}}^\top \underline{\mathcal{G}}^e \underline{\mathbf{N}} dx \end{array} \right. \quad (3.40a)$$

$$\left\{ \begin{array}{l} \underline{\mathbf{K}}^e = \int_0^l \underline{\mathbf{B}}^\top \underline{\mathcal{D}}^e \underline{\mathbf{B}} dx \end{array} \right. \quad (3.40b)$$

$$\left\{ \begin{array}{l} \underline{\mathbf{M}}^e = \int_0^l \underline{\mathbf{N}}^\top \underline{\mathcal{M}}^e \underline{\mathbf{N}} dx \\ \underline{\mathbf{G}}^e = \int_0^l \underline{\mathbf{N}}^\top \underline{\mathcal{G}}^e \underline{\mathbf{N}} dx \\ \underline{\mathbf{K}}^e = \int_0^l \underline{\mathbf{B}}^\top \underline{\mathcal{D}}^e \underline{\mathbf{B}} dx \end{array} \right. \quad (3.40c)$$

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

$$\underline{\mathbf{M}}^e \ddot{\vec{\mathbf{q}}} + \Omega \underline{\mathbf{G}}^e \dot{\vec{\mathbf{q}}} + \underline{\mathbf{K}}^e \vec{\mathbf{q}} = 0 \quad (3.41)$$

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.

3.1.1.6 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. This will allow for the study of the destabilizing effect in general, and the factors that may contribute stability such as structural damping and anisotropy of supports. Another area of interest is the design of components for specific rotor geometry to maximize the stability in the system. This stability analysis is only possible with the inclusion of some destabilizing force, [16],[15],[23],[39].

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 of the shaft. Obviously this is easier to define in the rotating reference frame, as the variables in this coordinate system directly represent the flex of the shaft from its neutral position. This relationship is as follows:

$$\vec{F}_{\xi\eta} = -c_r \begin{Bmatrix} \dot{\xi}_c \\ \dot{\eta}_c \end{Bmatrix} \quad (3.42)$$

Now to translate this force back to the stationary reference frame, we will see the rotation transformation matrix

$$\underline{\mathcal{R}} = \begin{bmatrix} \cos \Omega t & \sin \Omega t \\ -\sin \Omega t & \cos \Omega t \end{bmatrix} \quad (3.43)$$

transforms stationary into rotating coordinates

$$\begin{cases} \begin{Bmatrix} \xi_c \\ \eta_c \end{Bmatrix} = \underline{\mathcal{R}} \begin{Bmatrix} y_c \\ z_c \end{Bmatrix} \\ \begin{Bmatrix} \dot{\xi}_c \\ \dot{\eta}_c \end{Bmatrix} = \underline{\mathcal{R}} \begin{Bmatrix} \dot{y}_c \\ \dot{z}_c \end{Bmatrix} + \dot{\underline{\mathcal{R}}} \begin{Bmatrix} y_c \\ z_c \end{Bmatrix} \end{cases} \quad (3.44)$$

where,

$$\dot{\underline{\mathcal{R}}} = \Omega \begin{bmatrix} -\sin \Omega t & \cos \Omega t \\ -\cos \Omega t & -\sin \Omega t \end{bmatrix} \quad (3.45)$$

substituting the second equation in 3.44 for the velocities in 3.42

$$\vec{\mathcal{F}}_{xy} = -c_r \begin{Bmatrix} \dot{y}_c \\ \dot{z}_c \end{Bmatrix} - c_r \Omega \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \begin{Bmatrix} y_c \\ z_c \end{Bmatrix} \quad (3.46)$$

From the equation (3.46) we see a dependence on both velocity and position. The portion dependent on the velocity is inherently stable as it 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 structural damping forces, the system will destabilize.

For the beam element, the constitutive relationship[39] is comprised of both viscous and hysteretic forms of damping, η_v & η_h respectively.

$$\sigma_{xx} = E \left\{ \frac{\epsilon_{xx}}{\sqrt{1 + \eta_h^2}} + \left(\eta_v + \frac{\eta_h}{\omega \sqrt{1 + \eta_h^2}} \right) \dot{\epsilon}_{xx} \right\} \quad (3.47)$$

through the use of kinematics to obtain strain-displacement relations,

$$\begin{cases} \epsilon_{xx} = -r \cos(\Omega - \omega)t \frac{\partial^2 R}{\partial x^2} \\ \dot{\epsilon}_{xx} = (\Omega - \omega)r \sin(\Omega - \omega)t \frac{\partial^2 R}{\partial x^2} - r \cos(\Omega - \omega)t \frac{\partial}{\partial t} \frac{\partial^2 R}{\partial x^2} \end{cases} \quad (3.48)$$

and inspection to obtain moment equations,

$$\begin{cases} M_y = \int_0^{2\pi} \int_0^a [w + r \sin \Omega t] \sigma_{xx} dr (rd(\Omega t)) \\ M_z = \int_0^{2\pi} \int_0^a [-v + r \cos \Omega t] \sigma_{xx} dr (rd(\Omega t)) \end{cases} \quad (3.49)$$

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

$$\begin{Bmatrix} M_y \\ M_z \end{Bmatrix} = EI \begin{bmatrix} \eta_a & \Omega \eta_v + \eta_b \\ \Omega \eta_v + \eta_b & -\eta_a \end{bmatrix} \begin{Bmatrix} v'' \\ w'' \end{Bmatrix} + EI \begin{bmatrix} \eta_v & 0 \\ 0 & -\eta_v \end{bmatrix} \begin{Bmatrix} \dot{v}'' \\ \dot{w}'' \end{Bmatrix} \quad (3.50)$$

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

Now use the same strategy followed when solving for the weak form of the beam differential equations using the Principle of Virtual Displacements starting at equation (3.27). Then use the separation of variables of defined by equation (3.31) to arrive at the total beam element equations of motion including internal damping as

$$\underline{\mathbf{M}}^e \ddot{\vec{\mathbf{q}}} + (\eta_v \underline{\mathbf{K}}^e + \Omega \underline{\mathbf{G}}^e) \dot{\vec{\mathbf{q}}} + [\eta_a \underline{\mathbf{K}}^e + (\Omega \eta_v + \eta_b) \underline{\mathbf{C}}^e] \vec{\mathbf{q}} = 0 \quad (3.51)$$

where,

$$\underline{\mathcal{I}} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad \& \quad \underline{\mathbf{C}}^e = \int_0^l \underline{\mathbf{B}}^T \underline{\mathcal{I}} \underline{\mathcal{D}} \underline{\mathbf{B}} dx \quad (3.52)$$

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

3.1.1.7 Beam Element in Complex Coordinates

Complex coordinates collapse the equations of each plane into one set of equations. This lends properties to 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, their condensation has no effect on the remainder of system matrices. Complex coordinates used here are defined as:

$$\vec{\mathbf{s}} = \begin{Bmatrix} \vec{r} \\ \vec{p} \end{Bmatrix} = \begin{Bmatrix} v + iw \\ \theta - i\psi \end{Bmatrix} \quad (3.53)$$

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

$$\underline{\mathbf{N}}^c = \begin{bmatrix} T_{t_1y} & T_{r_1y} & T_{t_2y} & T_{r_2y} \\ R_{t_1} & R_{r_1} & R_{t_2} & R_{r_2} \end{bmatrix} \quad (3.54)$$

in

$$\vec{s} = \underline{\mathbf{N}}^c \vec{\mathbf{q}}^c \quad (3.55)$$

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

To convince this property of the system in the complex plane, a proof for the elemental equations of motion will be made. Taking equation (3.24b), adding equation (3.24c) 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

$$\kappa G A (\vec{r}'' - \vec{p}') = \rho A \ddot{\vec{r}} \quad (3.56)$$

and taking equation (3.24e) and subtracting equation (3.24d) multiplied by i gives the rotation equation in the complex plane

$$EI\vec{p}'' - \kappa G A (\vec{r}' - \vec{p}) = \rho I \ddot{\vec{p}} + i\rho J \Omega \dot{\vec{p}} \quad (3.57)$$

which by inspection of both of these equations it is evident that the form is the same as equations (3.24b) and (3.24e) 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 (3.51) are

$$\underline{\mathbf{M}}^{ec} \ddot{\vec{\mathbf{q}}}^c + (\eta_v \underline{\mathbf{K}}^{ec} - i\Omega \underline{\mathbf{G}}^{ec}) \dot{\vec{\mathbf{q}}}^c + [\eta_a \underline{\mathbf{K}}^{ec} - i(\Omega\eta_v + \eta_b) \underline{\mathbf{C}}^{ec}] \vec{\mathbf{q}}^c = 0 \quad (3.58)$$

3.2 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

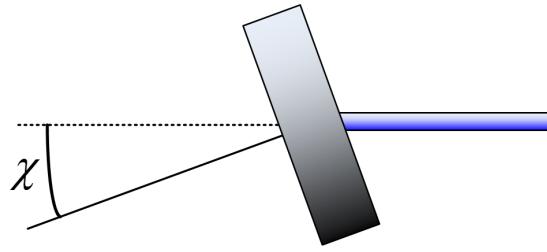


Figure 3.6: Depiction of skew angle χ

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.

$$\underline{\mathbf{M}}^d \ddot{\underline{\mathbf{q}}}_k + \Omega \underline{\mathbf{G}}^d \dot{\underline{\mathbf{q}}}_k = \Omega^2 \vec{\mathbf{F}}^d \quad (3.59)$$

the superscript d 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: $\vec{\mathbf{q}}_k = [v, w, -\psi, \theta, u, \phi]^T$. The matrices and forcing array of (5.6) are as follows,

$$\left\{ \begin{array}{l} \underline{\mathbf{M}}^d = \begin{bmatrix} \rho Al & 0 & 0 & 0 & 0 & 0 \\ 0 & \rho Al & 0 & 0 & 0 & 0 \\ 0 & 0 & \rho I_z & 0 & 0 & 0 \\ 0 & 0 & 0 & \rho I_y & 0 & 0 \\ 0 & 0 & 0 & 0 & \rho Al & 0 \\ 0 & 0 & 0 & 0 & 0 & \rho J_x \end{bmatrix} \quad \underline{\mathbf{G}} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \rho J_x & 0 & 0 \\ 0 & 0 & -\rho J_x & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \\ \vec{\mathbf{F}}^d = \begin{Bmatrix} \rho Ale \cos(\Omega t + \delta_\varepsilon) \\ \rho Ale \sin(\Omega t + \delta_\varepsilon) \\ -\rho(I_y - J_x)\chi \sin(\Omega t) \\ \rho(I_z - J_x)\chi \cos(\Omega t) \\ 0 \\ 0 \end{Bmatrix} \end{array} \right\} \quad (3.60)$$

Disk unbalance is caused by an eccentricity, or a geometrical 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 moment unbalance forces, the third and fourth equations of $\vec{\mathbf{F}}^d$ in (3.60), are caused by the skew angle, χ , which is the angle the disk major axis forms with the axis of rotation as in Figure 3.6. The major axis is the axis normal from the disk face from which the polar moment of inertia, J_x is defined.

3.2.1 Disk in complex coordinates

Disk equations of motion (5.6) are converted to the complex plane in the same manner the beam element was derived in §3.1.1.7

$$\underline{\mathbf{M}}^{dc} \ddot{\underline{\mathbf{q}}}^c_k - i\Omega \underline{\mathbf{G}}^{dc} \dot{\underline{\mathbf{q}}}^c_k = \Omega^2 \vec{\mathbf{F}}^{dc} \quad (3.61)$$

where,

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

3.3 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

$$\underline{\mathbf{D}}^b \dot{\underline{\mathbf{q}}}_k + \underline{\mathbf{K}}^b \vec{\mathbf{q}}_k = 0 \quad (3.62)$$

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

$$\underline{\mathbf{K}}^b = \begin{bmatrix} k_{yy} & k_{yz} & 0 & 0 & 0 & 0 \\ k_{zy} & k_{zz} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad \underline{\mathbf{D}}^b = a\underline{\mathbf{K}}^b \quad (3.63)$$

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.

3.3.1 Bearing in Complex Coordinates

Following the same methodology as sections §3.1.1.7 & §3.2.1, the equation of motion in the complex plane is

$$\underline{\mathbf{D}}^{bc} \dot{\underline{\mathbf{q}}}^c_k + \underline{\mathbf{K}}^{bc} \underline{\mathbf{q}}^c_k = 0 \quad (3.64)$$

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.

3.4 Assembly of the Global Systems 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.

3.4.1 In the Real Coordinate System

$$\underline{\mathbf{M}}\ddot{\underline{\mathbf{q}}} + \underline{\mathbf{G}}\dot{\underline{\mathbf{q}}} + \underline{\mathbf{K}}\underline{\mathbf{q}} = \Omega^2 \vec{\mathbf{F}} \quad (3.65)$$

Each summation is defined as

$$\begin{cases} \underline{\mathbf{M}} = \underline{\mathbf{M}}_G^e + \underline{\mathbf{M}}_G^b + \underline{\mathbf{M}}_G^d & \underline{\mathbf{D}} = \eta_v \underline{\mathbf{K}}_G^e + \Omega \underline{\mathbf{G}}_G^e + \Omega \underline{\mathbf{G}}_G^d + \underline{\mathbf{D}}_G^b \\ \underline{\mathbf{K}} = \eta_a \underline{\mathbf{K}}_G^e + (\Omega \eta_v + \eta_b) \underline{\mathbf{C}}_G^e + \underline{\mathbf{K}}_G^b & \vec{\mathbf{F}} = \vec{\mathbf{F}}_G^d \end{cases}$$

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.

3.4.2 In the Complex Coordinate System

$$\underline{\mathbf{M}}\ddot{\underline{\mathbf{q}}}^c + \underline{\mathbf{D}}\dot{\underline{\mathbf{q}}}^c + \underline{\mathbf{K}}\underline{\mathbf{q}}^c = \Omega^2 \vec{\mathbf{F}} \quad (3.66)$$

where,

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

3.5 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 (3.65) & (3.66). Many techniques exist to provide frequency and time domain information on the solution of this system of equations. In their current state, equations (3.65) & (3.66) may be analyzed using frequency domain analysis techniques to be discussed in §4. 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 §2. This time domain method of processing the FEM model is vital in analyzing non-linear differential equations that may result from a more detailed model of bearings, asymmetry, and many other effects.

Chapter 4

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.

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 necessity to ramp up quickly in speed.

For a simple critical speed computation, assuming the form of a homogeneous solution $\vec{q} = e^{i\Omega t}$ while neglecting all damping and forcing in the equations of motion will provide the dynamic undamped forced whirling matrix. Exploring the eigenvalues 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 $\vec{q} = e^{st}$ is assumed so that the 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

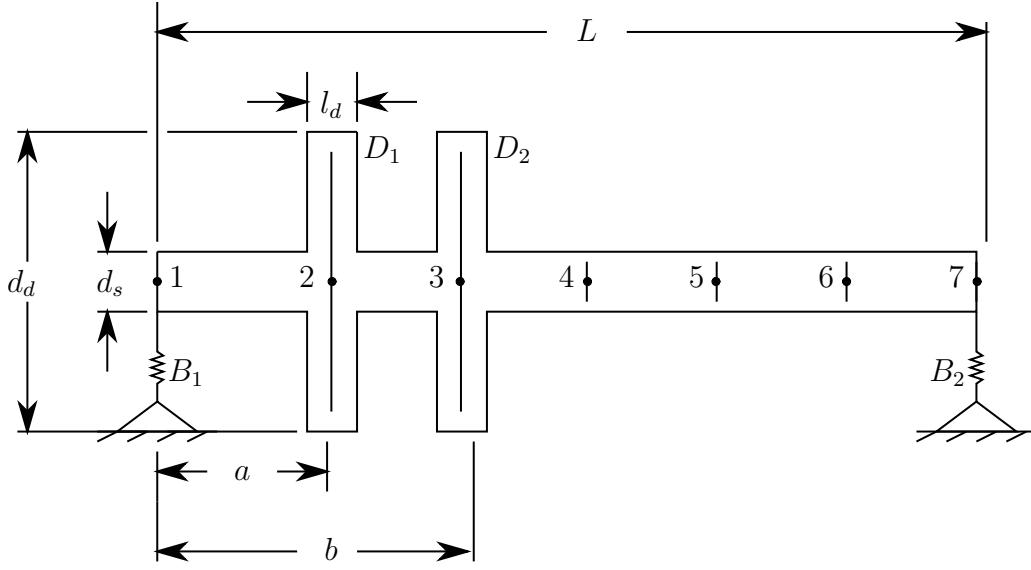


Figure 4.1: Diagram of Two disk model example problem.

damping, and the imaginary is damped natural frequencies of the system. Particular integrals of the form $\vec{q} = e^{i\omega t}$ & $\vec{q} = e^{i\Omega t}$ 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 4.1. The geometry and material properties are listed in Table 4.1. The rotor is discretized as in figure 4.1 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.

4.1 State Space Representation and the Eigenvalue Problem

The system can be represented in state space where,

$$\dot{\vec{z}} = \underline{\mathbf{A}}\vec{z} + \underline{\mathbf{B}}\vec{w}, \quad \vec{z} = \begin{Bmatrix} \dot{\vec{q}} \\ \vec{q} \end{Bmatrix} \quad (4.1)$$

Table 4.1: Properties of disks, shaft elements, and bearings of the example problem.

	$\rho \left[\frac{kg}{m^3} \right]$	$d_s [m]$	ν	$E [Pa]$	$\eta_v [s]$	η_h
Shaft	7850	0.1	0.3	210×10^9	0.0002	0
	$\rho \left[\frac{kg}{m^3} \right]$	$d_d [m]$	$l_d [m]$			
Disks; D_1, D_2	7850	0.6	0.1			
	$k_x \left[\frac{N}{m} \right]$	$k_y \left[\frac{N}{m} \right]$	$c_x \left[\frac{Ns}{m} \right]$	$c_y \left[\frac{Ns}{m} \right]$		
Bearings; B_1, 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 $\underline{\mathbf{A}}$ & $\underline{\mathbf{B}}\vec{\mathbf{w}}$

$$\underline{\mathbf{A}} = \begin{bmatrix} -\underline{\mathbf{M}}^{-1}\underline{\mathbf{D}} & -\underline{\mathbf{M}}^{-1}\underline{\mathbf{K}} \\ \mathbf{I} & \mathbf{0} \end{bmatrix} \quad \& \quad \underline{\mathbf{B}}\vec{\mathbf{w}} = \Omega^2 \begin{Bmatrix} \underline{\mathbf{M}}^{-1}\vec{\mathbf{F}} \\ 0 \end{Bmatrix} \quad (4.2)$$

This equation is valid in both the complex and real coordinate plane. With the use of complex coordinates, the state vector is represented by $\vec{\mathbf{z}} = [\dot{\vec{\mathbf{q}}}^c, \vec{\mathbf{q}}^c]^\top$. \mathbf{I} & $\mathbf{0}$ 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 (4.1) and assuming a solution of $\vec{\mathbf{z}} = \vec{\Theta}e^{st}$ while neglecting the forcing term $\underline{\mathbf{B}}\vec{\mathbf{w}}$ leads to the eigenvalue problem

$$(\underline{\mathbf{A}} - s)\vec{\Theta} = \mathbf{0} \quad (4.3)$$

$\vec{\Theta}$ 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

$$\vec{\Theta} = \begin{Bmatrix} \vec{\theta}_{\dot{\vec{q}}} \\ \vec{\theta}_{\vec{q}} \end{Bmatrix} = \begin{Bmatrix} s\vec{\theta}_{\vec{q}} \\ \vec{\theta}_{\vec{q}} \end{Bmatrix}$$

4.2 Dynamic Response

The dynamic response of the system to unbalance can be achieved by substituting the particular integral $\vec{q} = \vec{q}_0 e^{i\Omega t}$ into the equation of motion (3.65),(3.66).

$$\begin{cases} \vec{q}_0 = (-\Omega^2 \underline{\mathbf{M}} + i\Omega \underline{\mathbf{D}} + \underline{\mathbf{K}})^{-1} \Omega^2 \vec{\mathbf{F}} \\ \vec{q}_0 = H(\Omega) \end{cases} \quad (4.4)$$

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{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 §3.2 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 $\vec{q} = \vec{q}_0 e^{i\omega t}$ 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 §4 the unbalance response to a disk b unbalance of $2 \times 10^{-5}[m]$ is shown in Figure 4.2. Note that the phase lag angle and the amplitude of the response are plotted side by side forming the Bode diagram. For this

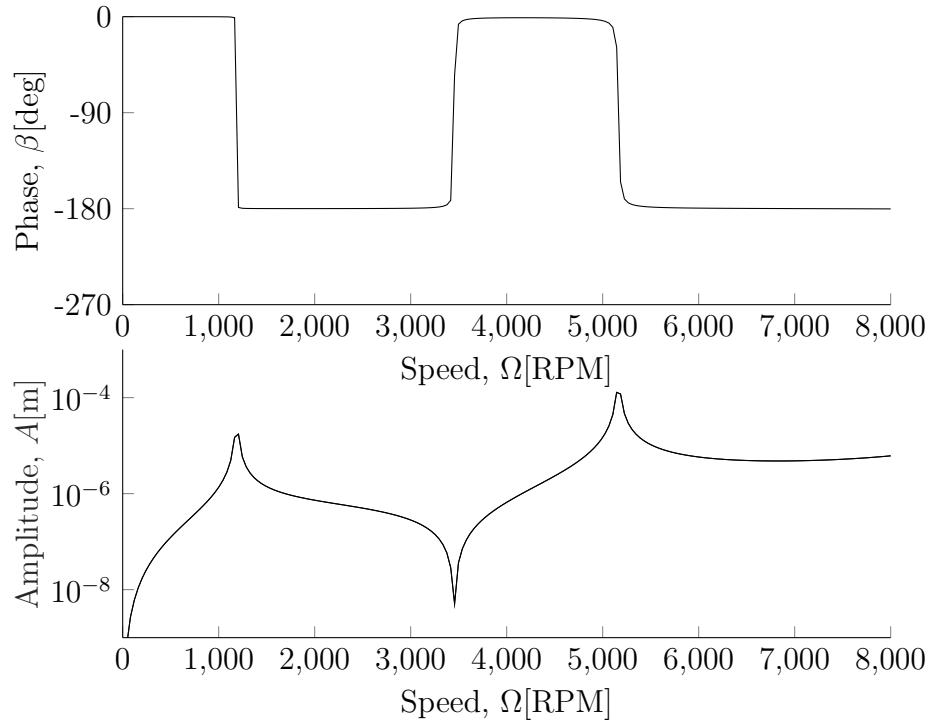
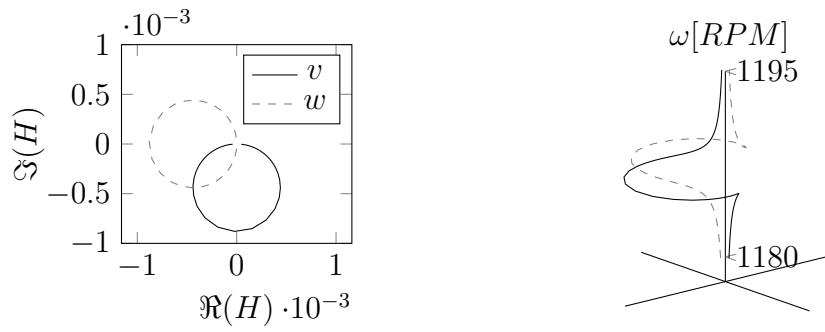
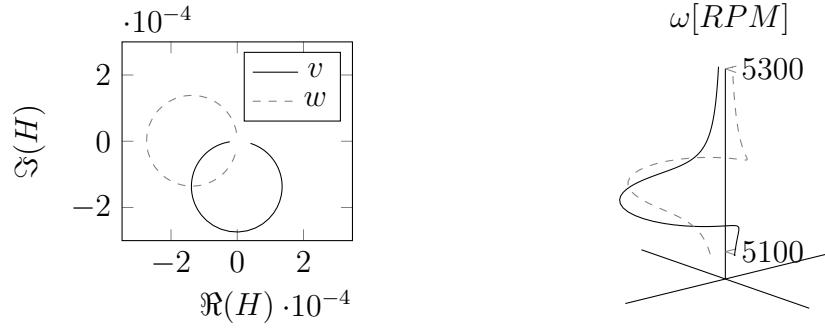


Figure 4.2: Bode diagram of the second disk subject to an unbalance at the second disk. Amplitudes of v & w are identical for this antisymmetric system, phase of w would be lagging v by 90[deg].

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 4.3 contains a separate 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 $\vec{q} = \vec{q}_0 e^{i\omega t}$ 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 ??) 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 4.3: Nyquist plots for the first two modes at node 6 for the example two disk problem.

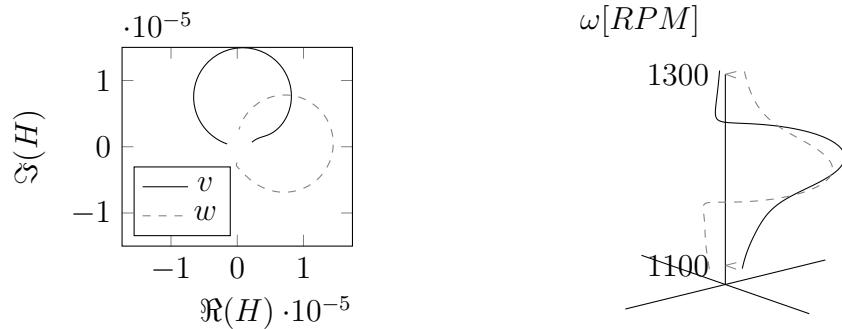


Figure 4.4: Nyquist Diagram for the first mode at node 6 in the whirl speed range $1100 < \omega < 1300[RPM]$ at a spin speed, $\Omega = 4000[RPM]$. Counter-clockwise path indicates instability.

4.3 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 §3.1.1.6, 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 §3.1.1.6 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 (4.3), 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 (4.3) on the example problem defined in §4 a plot of the roots locus can be found as in figure 4.5. 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

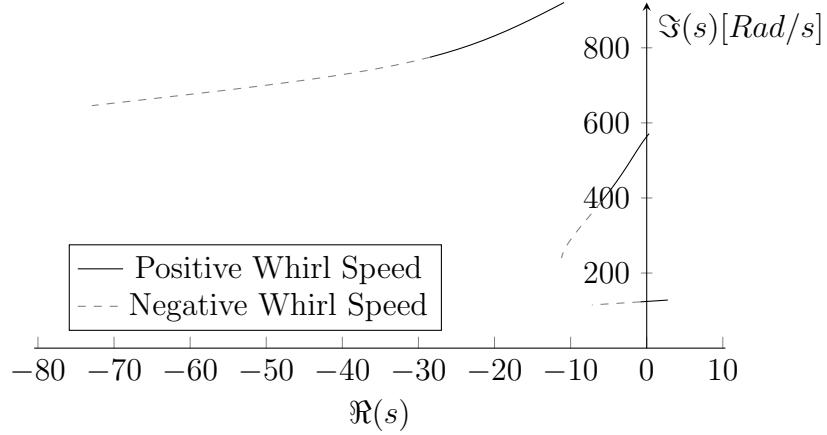


Figure 4.5: Roots Locus of the example problem with an internal damping coefficient of 0.0002[s].

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 §3.1.1.6 assists in this trend. It is evident by looking at figure 4.5 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 §5 will be to provide enough non-rotating damping to render as many modes as possible over-damped.

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 4.6. A direct method of measuring the stability of the system is to observe only

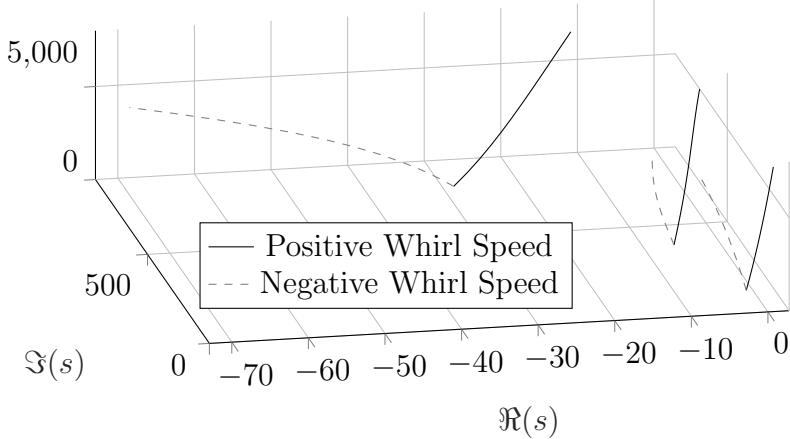


Figure 4.6: Roots Locus of the example problem in 3-D with an internal damping coefficient of 0.0002[s].

the real part of the eigenvalues of the dynamic matrix as they vary with spin speed. Since the solution was assumed to be of the form $\vec{z} = \vec{\Theta}e^{st}$, 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{z} gives: $\vec{z} = \vec{\Theta}e^{\sigma+i\omega_d t}$. 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 the ratio of the real part of the eigenvalue to the natural frequency as

$$\zeta = \frac{-\sigma}{\omega_n}, \quad \omega_n = \sqrt{\sigma^2 + \omega_d^2} \quad (4.5)$$

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 4.7 for the example problem defined in §4.

Finally, the stability can be analyzed using the damping coefficients(ζ) plotted

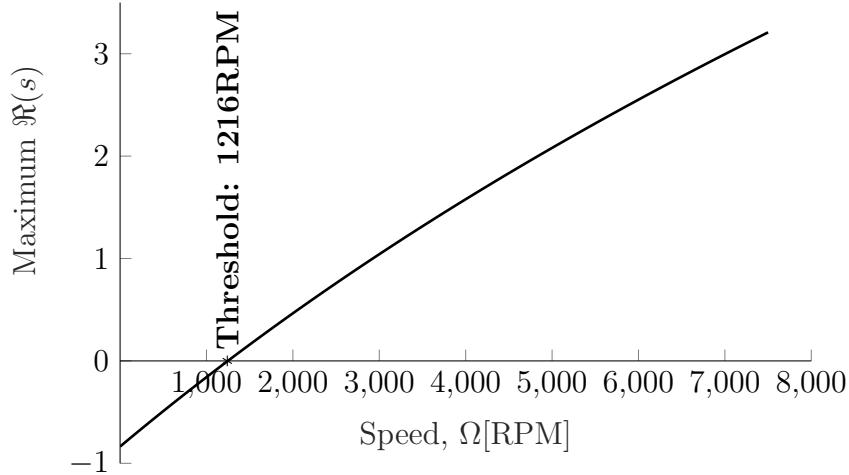


Figure 4.7: Stability plot of the example problem.

against speed. This figure, shown in 4.8, indicates instability as when ζ drops below zero.

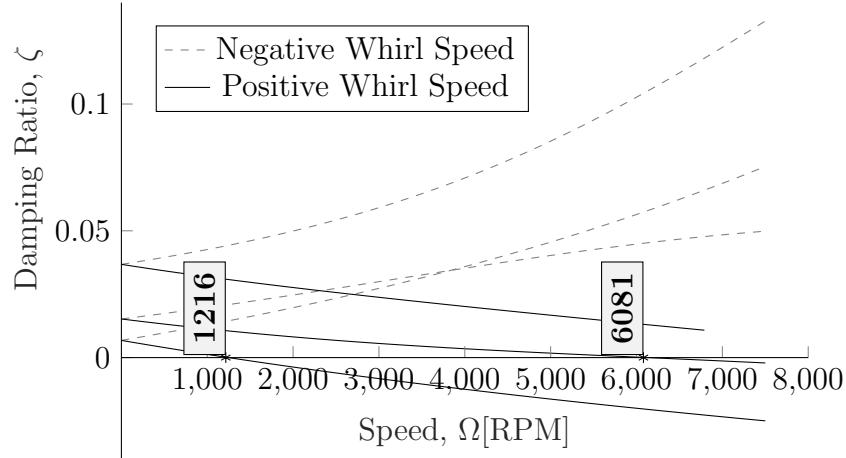


Figure 4.8: Damping ratio of the first three modes of the example problem, with indication of threshold of stability for each mode.

4.4 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 4.3. Spin speed is varied while calculating all of the critical whirl

speeds at each step of spin speed. The campbell diagram for the example problem is given in figure 4.9. 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 §4.5

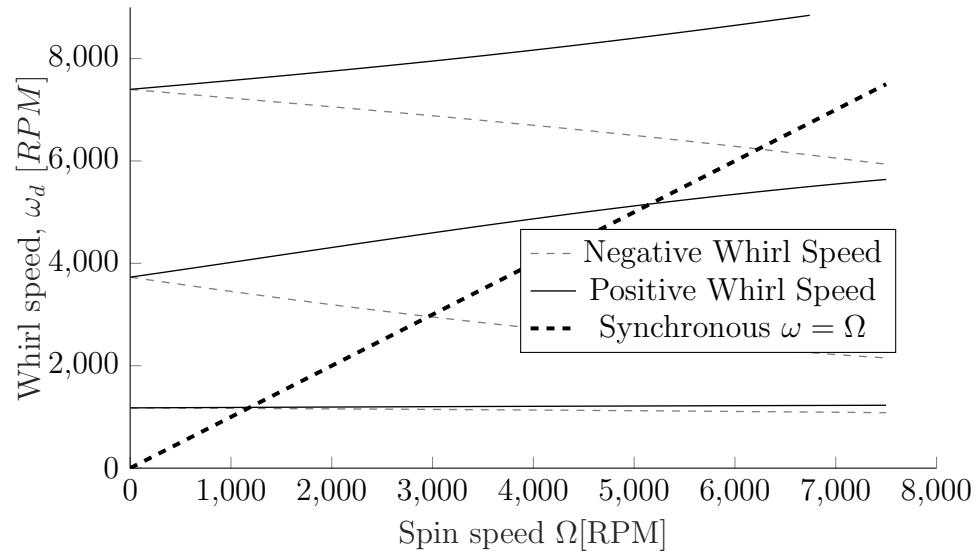


Figure 4.9: Campbell Diagram of the example problem.

4.5 Shapes

Deformed shape of each mode can be predicted using the eigenvectors of the eigenvalue problem from equation 4.3. 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 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 $\dot{\vec{q}}$, & \vec{q} the eigenvector will contain $2N$ modes. For a general i th mode of vibration, the eigenvector $\vec{\Theta}_i$ contains the displacement information. But, since the portion of $\vec{\Theta}_i$ pertaining to velocities, $\vec{\theta}_{\dot{q}i}$, is a linear combination of $\vec{\theta}_{\vec{q}i}$, all of the information is contained in one of these arrays. Therefore, the interpolation of displacements for the i th mode is given by

$$\vec{u}_i = \underline{\mathbf{N}} \vec{\theta}_{\vec{q}i} \quad (4.6)$$

where, the value of \vec{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 i th 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 4.10.

Shape figures for the first 3 modes of the example problem are presented in Figures 4.10a,4.10b, and 4.10c. 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 .

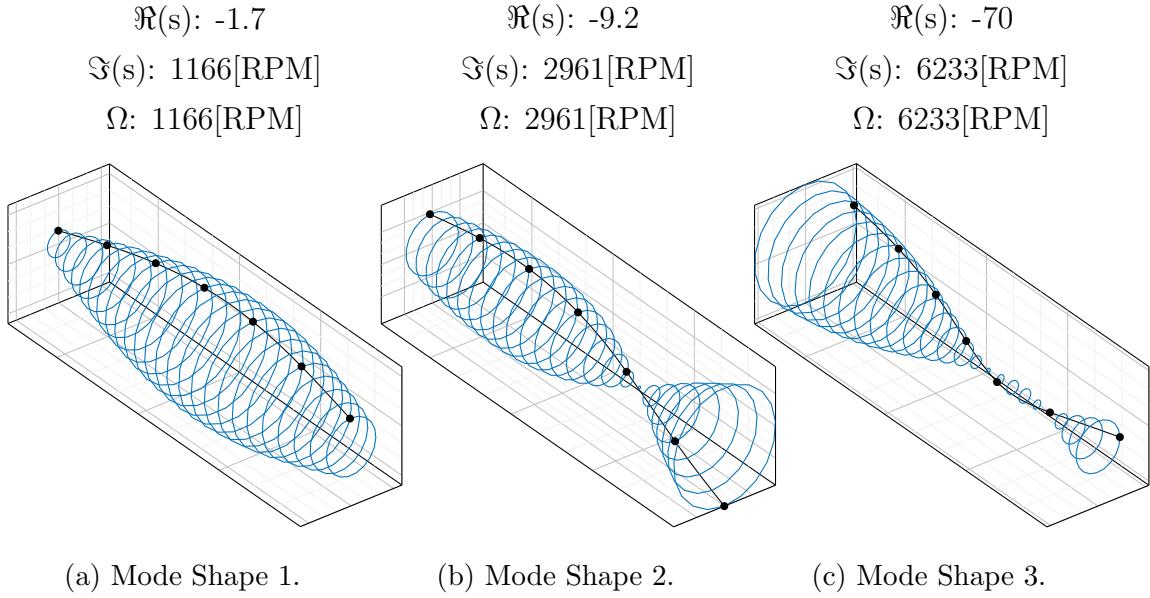


Figure 4.10: Modal shapes of the example problem.

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 4.10 it is evident that in the first mode, Figure 4.10a, 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 4.10a, 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 4.10b & 4.10c 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 5

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 §2.2, 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.

5.1 Physical System Description

The rotor system of interest is depicted in Figure 5.1. Geometric parameters are listed in Table 5.1. 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.

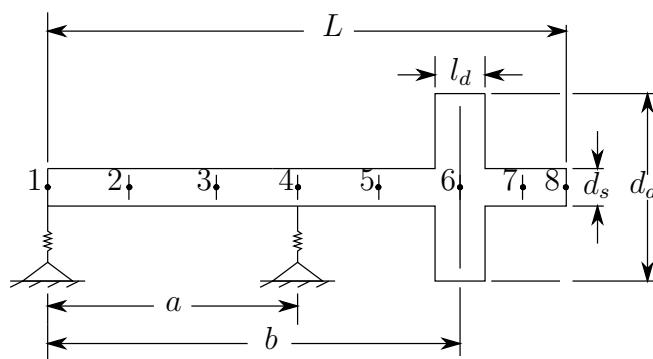


Figure 5.1: Overhung rotor system diagram.

Table 5.1: 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

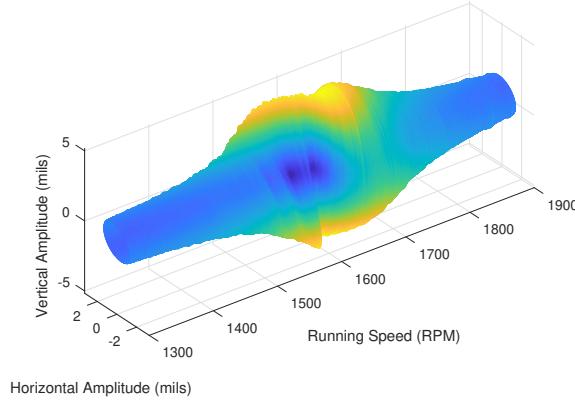


Figure 5.2: 3D Orbit of the experimental overhung rotor system.

5.2 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 5.2. 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 5.3, in which it is evident that synchronous vibration dominates the spectra. Though, for the production of the bode diagram (Figure 5.4) there was a benefit in clarity from filtering the data to synchronous speed.

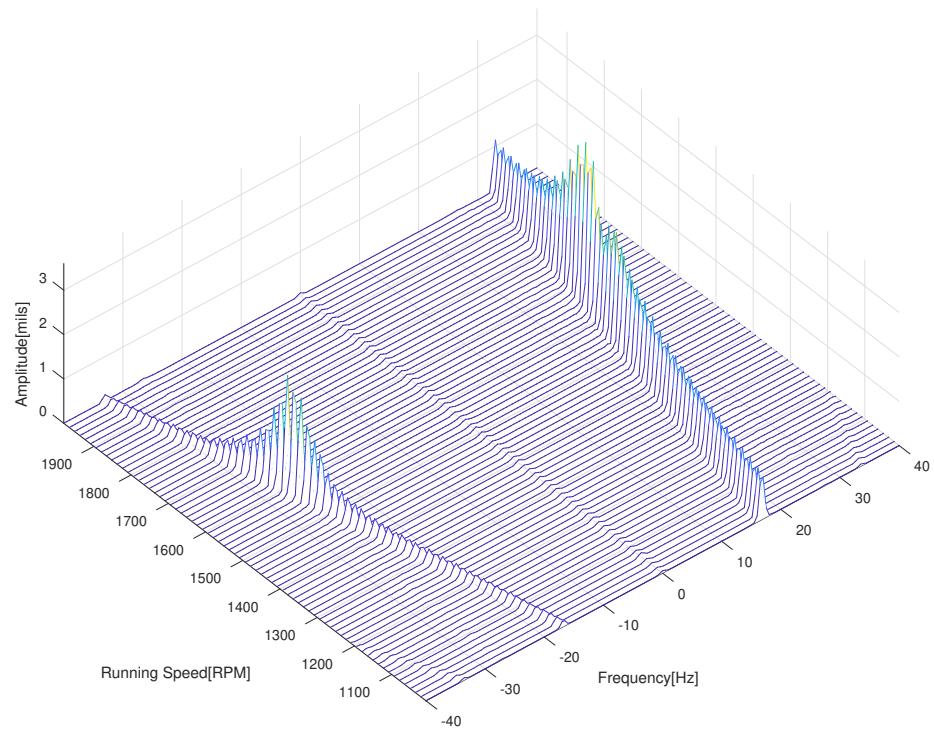


Figure 5.3: Cascade of the experimental overhung rotor system.

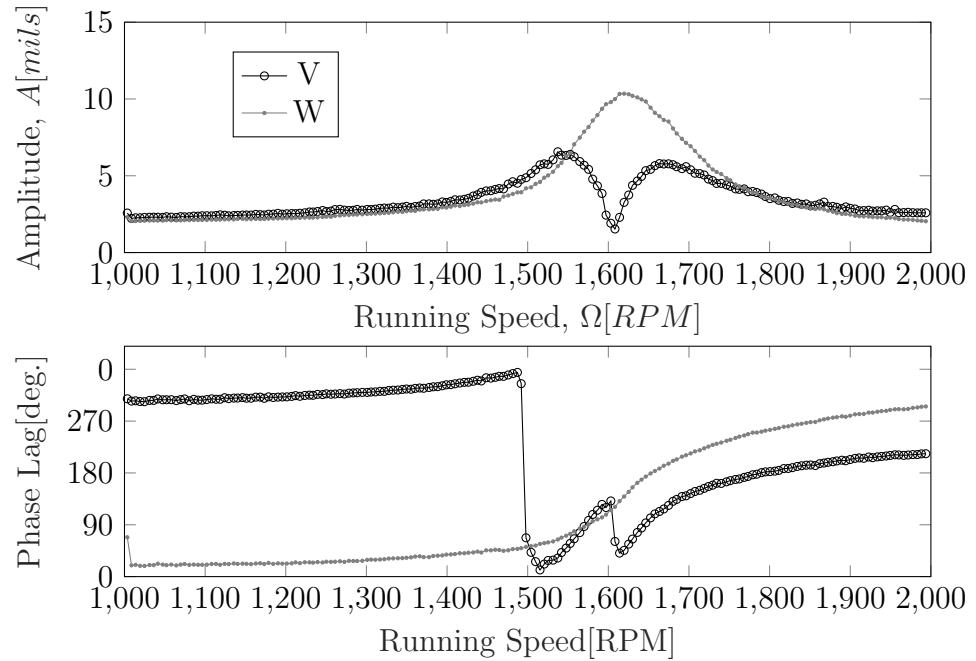


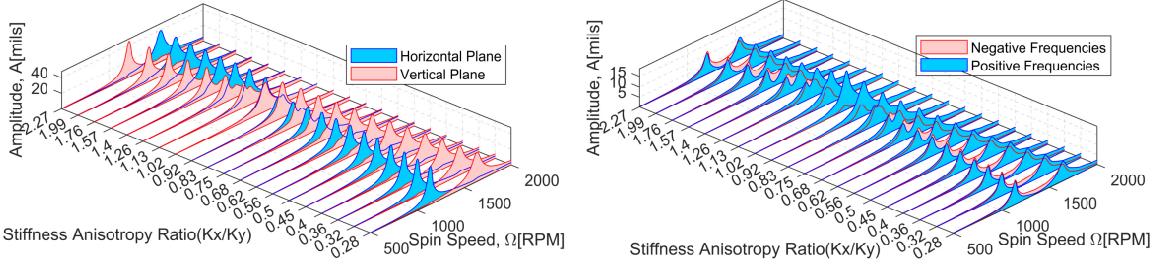
Figure 5.4: Bode diagram of the experimental overhung rotor filtered to 1X.

Table 5.2: 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		

5.3 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 5.1. 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 stiffness and damping to be derived in §5.3.1. Parameter values for the finite element model are provided in Table 5.2. 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 found natural frequency from the bode diagram (fig.5.4). 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 $2 \times 10^5 \left[\frac{N}{m} \right]$.



(a) 3D Orbit of the experimental overhung rotor system.
(b) 3D Orbit of the experimental overhung rotor system.

It is evident by inspection of the Bode diagram for the experimental system (fig.5.4), 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 5.3 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 5.5a&5.5b demonstrate this affect anisotropy has on both the real coordinates, as well as positive and negative whirl amplitudes.

Using the bode diagram, the stiffness anisotropy is adjusted until the shapes of the amplitudes and phases match the experimental results of figure 2.4. Then damping is added to the system to appropriately match the experimental results. The resulting bode diagram is Figure 5.6. 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]. The new Campbell diagram of Figure 5.7, the roots locus of Figure 5.8. Additionally, the first three mode shapes are plotted for this theoretical model (Figures 5.9a, 5.9b, & 5.9c). 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. 5.7, as the first mode natural frequency does not change significantly over the speed range. On the other hand, the second mode has its antinode at node 6 so the disk experience maximum gyroscopic moments and the second mode critical

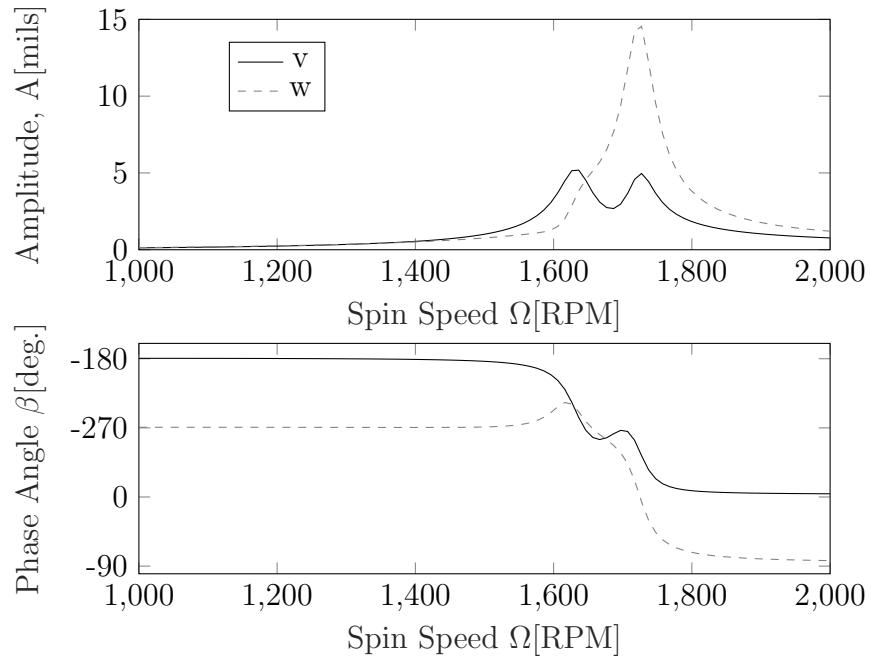


Figure 5.6: Bode Diagram of overhung system without AMB.

speed is much more dependent on speed.

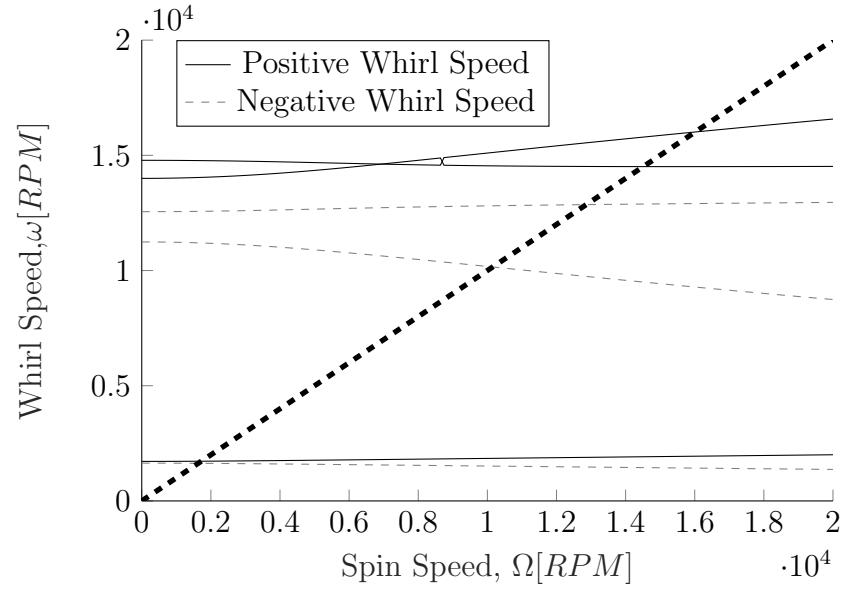


Figure 5.7: Campbell Diagram of the overhung system without AMB.

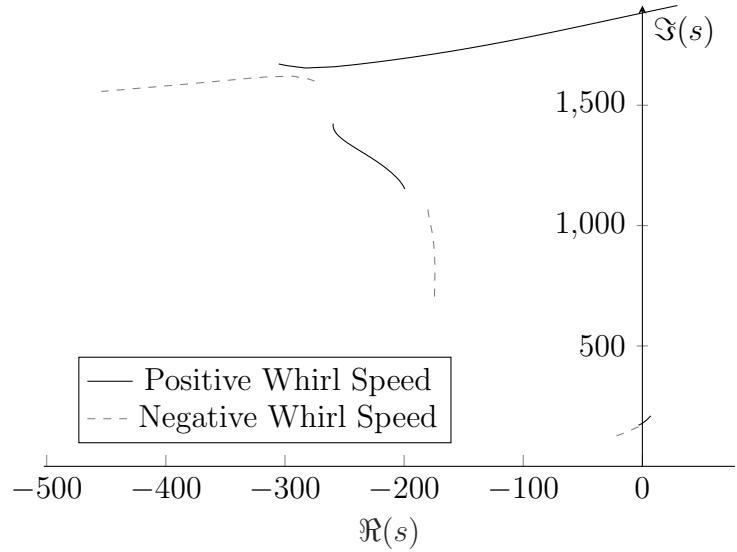


Figure 5.8: Roots Locus of overhung system without AMB.

5.3.1 Active Magnetic Bearing

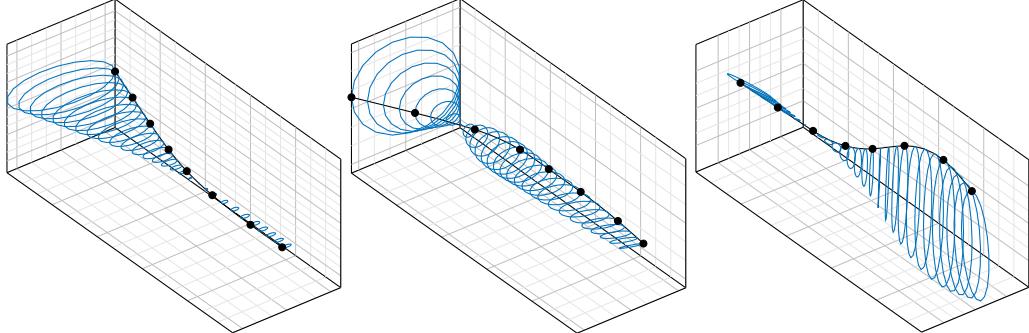
Magnetic pole-rotor relationship will be derived based on the detailed derivation in [9]. Assumptions made in this model are:

- 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

$$F_m = \frac{-k_m i^2}{l_g^2} \quad (5.1)$$

$\Re(s): -2.6,$	$\Re(s): -1.5e+02,$	$\Re(s): -3.7e+02,$
$\Im(s): 1633[\text{RPM}]$	$\Im(s): 1.015e+04[\text{RPM}]$	$\Im(s): 1.451e+04[\text{RPM}]$
$\Omega: 1633[\text{RPM}]$	$\Omega: 1.015e+04[\text{RPM}]$	$\Omega: 1.451e+04[\text{RPM}]$



(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 4 total magnets in this model, 2 in each direction, and 8 total poles where 2 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

$$F_y = K_m \left\{ \left(\frac{i_0 + i_y}{g_0 + v} \right)^2 - \left(\frac{i_0 - i_y}{g_0 - v} \right)^2 \right\} \quad \& \quad F_z = K_m \left\{ \left(\frac{i_0 + i_z}{g_0 + w} \right)^2 - \left(\frac{i_0 - i_z}{g_0 - w} \right)^2 \right\} \quad (5.2)$$

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 (5.2), while assuming the operating point for the system is where all positions and control currents are zero, results in

$$F_y = k_i i_y + k_y v, \quad \& \quad F_z = k_i i_z + k_z w \quad (5.3)$$

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

5.3.1.1 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

$$i_y = -k_g(k_p v + k_v \dot{v}), \quad \& \quad i_z = -k_g(k_p w + k_v \dot{w}) \quad (5.4)$$

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

$$F_y = -(k_g k_i k_p - k_s)v - k_g k_i k_v \dot{v}, \quad \& \quad F_z = -(k_g k_i k_p - k_s)w - k_g k_i k_v \dot{w} \quad (5.5)$$

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

$$\underline{\mathbf{D}}^m \dot{\vec{\mathbf{q}}}_k + \underline{\mathbf{K}}^m \vec{\mathbf{q}}_k = 0 \quad (5.6)$$

where,

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

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 $v = (A/2)\omega$, 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, v , of $0.02[m/s]$. Looking also at the velocity 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 calculated 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. Since the total voltage supply of most digital to analog converters is limited to $10[V]$ 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 cap 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. The voltage should not need to peak above $1.25[V]$, 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.

5.4 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 5.10a. 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 5.3 as well as the control values whose determinations will be evaluated.

First the 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 conical mode 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 6. With both these pieces of information, node 7 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.

To determine best derivative control, the proportional control was set to zero and the derivative increased until the first mode on the Roots locus(figure 5.8) moved away from the imaginary axis, becoming more damped. The resulting movement of the Roots on the Roots Locus is given in Figure 5.11. Notice how none of the new modes that appear on the roots locus 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 5.10b. 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.

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 possible 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 does not help. So the optimal control is determined to be with $k_v = 10[Vs/m]$. The remaining parameters for this resulting controller are listed in table 5.3. It is worthwhile to note that this 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 5.12. Certainly it can be concluded that the AMB is successfully performing the desired task of reducing the vibration while also improving the stability of the system.

Now 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 parameters 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 5.13

Table 5.3: Active Magnetic Bearing Parameters.

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

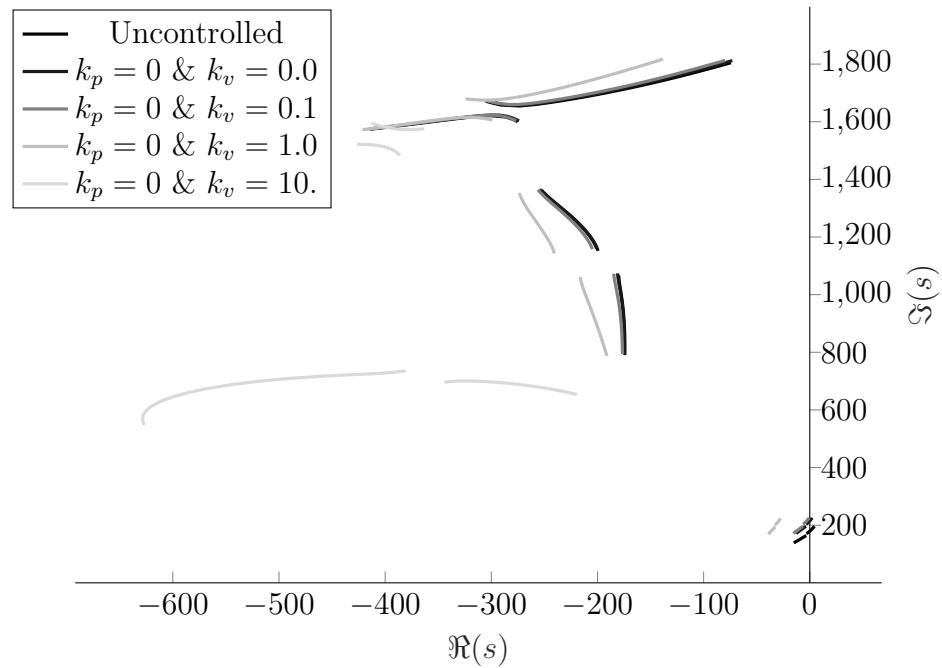
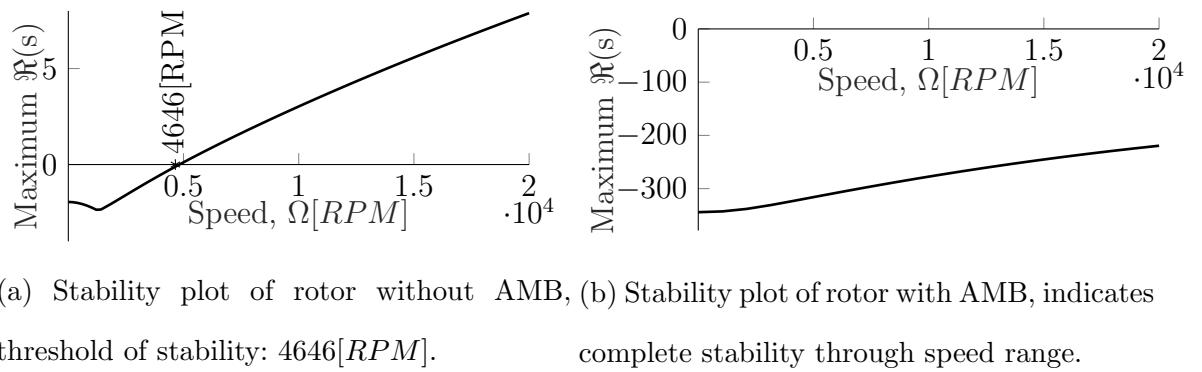


Figure 5.11: Roots locus of Overhung rotor system with varying k_v .

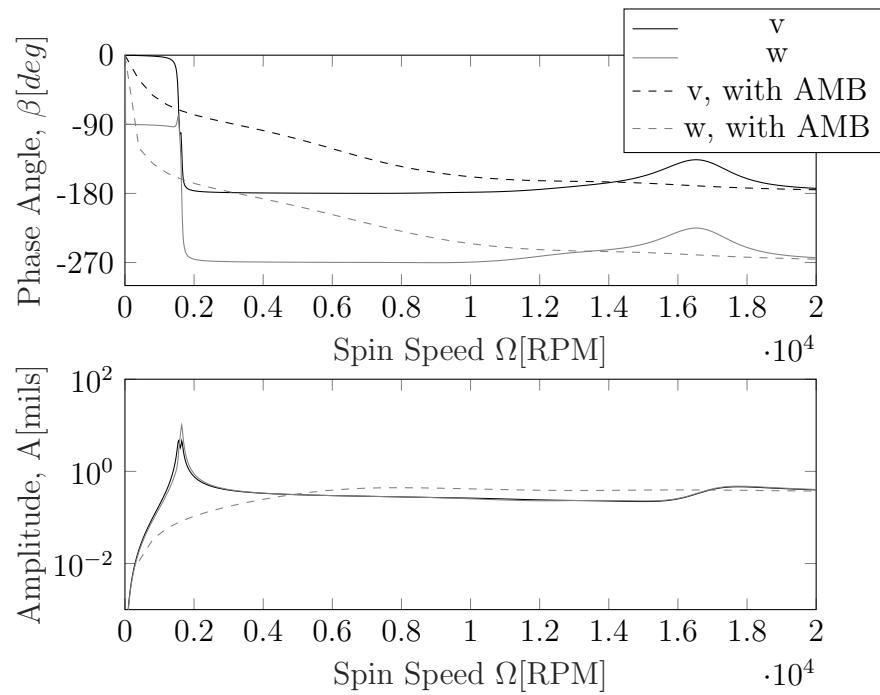


Figure 5.12: Bode diagram at node 6 comparing the rotor without AMB(solid) and with AMB(dashed).

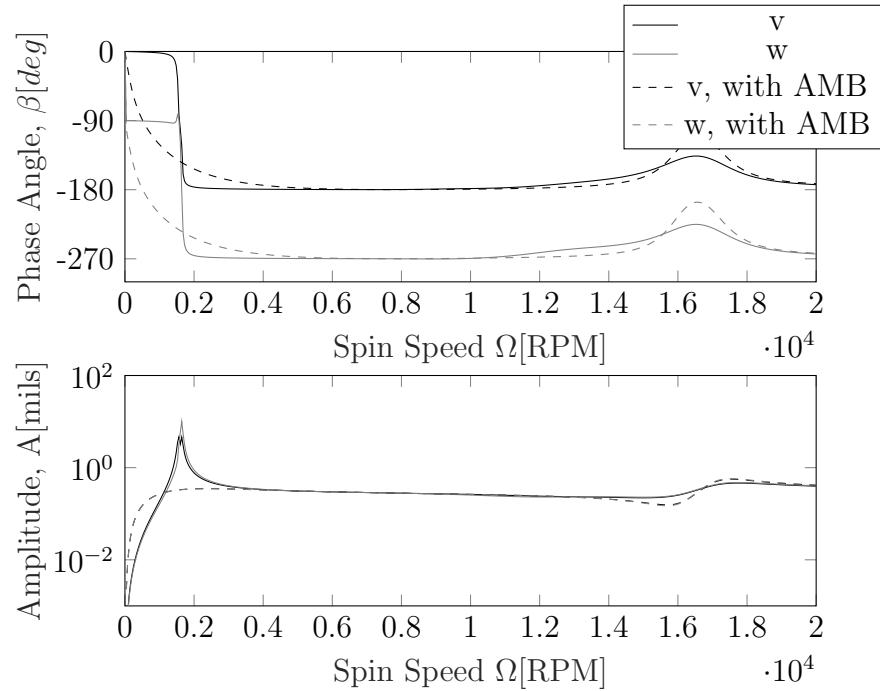


Figure 5.13: Bode diagram at node 6 comparing the rotor without AMB(solid) and with AMB(dashed) for complete levitation at node 4.

BIBLIOGRAPHY

- [1] M. L. Adams. *Rotating machinery vibration: from analysis to troubleshooting*. CRC Press, 2009.
- [2] L. Andersen and S. R. Nielsen. Elastic beams in three dimensions. *Aalborg University*, 2008.
- [3] G. Barbaraci. Axial active magnetic bearing design. *Journal of Vibration and Control*, 22(5):1190–1197, 2016.
- [4] D. E. Bently and T. Hatch'Charles. Fundamentals of rotating machinery diagnostics. *Mechanical Engineering-CIME*, 125(12):53–54, 2003.
- [5] A. Bouaziz, S. Bouaziz, T. Hentati, J. Y. Cholley, and M. Haddar. Vibrations monitoring of high speed spindle with active magnetic bearings in presence of defects. *International Journal of Applied Electromagnetics and Mechanics*, 49(2):207–221, 2015.
- [6] D. W. Childs. *Turbomachinery rotordynamics: phenomena, modeling, and analysis*. John Wiley & Sons, 1993.
- [7] R. R. Craig and A. J. Kurdila. *Fundamentals of structural dynamics*. John Wiley & Sons, 2006.
- [8] S. M. Darbandi, A. Habibollahi, M. Behzad, H. Salarieh, and H. Mehdigholi. Sensor runout compensation in active magnetic bearings via an integral adaptive observer. *Control Engineering Practice*, 48:111–118, 2016.
- [9] 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.

- [10] A. Dimarogonas and S. Haddad. Vibration for engineers. *VhS^ WI) IN914A R4 22k AWn 22kVAR5£ ilOuF 100k [^ 2k 22k 22k*, page 252, 1993.
- [11] 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.
- [12] M. G. Farmakopoulos, M. D. Thanou, P. G. Nikolakopoulos, C. A. Papadopoulos, and A. P. Tzes. A control model of active magnetic bearings. In *Proceedings of the 3rd International Conference of Engineering Against Failure (ICEAF III)*, pages 26–28, 2013.
- [13] 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.
- [14] G. Genta. Consistent matrices in rotor dynamic. *Meccanica*, 20(3):235–248, 1985.
- [15] 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.
- [16] G. Genta. *Dynamics of rotating systems*. Springer Science & Business Media, 2007.
- [17] T. Gmur and J. Rodrigues. Shaft finite elements for rotor dynamics analysis. *Journal of Vibration and Acoustics*, 113(4):482–493, 1991.
- [18] U. Goerguelue. Beam theories the difference between euler-bernoulli and timoschenko. *Lecture Handouts*, 2009.

- [19] P. Goldman and A. Muszynska. Application of full spectrum to rotating machinery diagnostics. *Orbit*, 20(1):17–21, 1999.
- [20] 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.
- [21] N. Instruments. Ni labview for compactrio developer’s guide. 2013.
- [22] C. Jin, Y. Xu, J. Zhou, and C. Cheng. Active magnetic bearings stiffness and damping identification from frequency characteristics of control system. *Shock and Vibration*, 2016, 2016.
- [23] M. A. Kandil. *On rotor internal damping instability*. PhD thesis, Imperial College London (University of London), 2005.
- [24] B. Kirchgässner. Finite elements in rotordynamics. *Procedia Engineering*, 144:736–750, 2016.
- [25] A. Kuperman. Uncertainty and disturbance estimator-assisted control of a two-axis active magnetic bearing. *Transactions of the Institute of Measurement and Control*, 38(6):764–772, 2016.
- [26] Y. Luo. An efficient 3d timoshenko beam element with consistent shape functions. *Adv. Theor. Appl. Mech*, 1(3):95–106, 2008.
- [27] A. Muszynska. *Rotordynamics*. CRC press, 2005.
- [28] V. Mykhaylyshyn. *Application of Active Magnetic Force Actuator for Control of Flexible Rotor System Vibrations*. PhD thesis, Cleveland State University, 2011.
- [29] H. Nelson. A finite rotating shaft element using timoshenko beam theory. *Journal of mechanical design*, 102(4):793–803, 1980.

- [30] H. Nelson and J. McVaugh. The dynamics of rotor-bearing systems using finite elements. *Journal of Engineering for Industry*, 98(2):593–600, 1976.
- [31] 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.
- [32] H. Roy, A. Das, and J. Dutt. An efficient rotor suspension with active magnetic bearings having viscoelastic control law. *Mechanism and Machine Theory*, 98:48–63, 2016.
- [33] G. Schweitzer. Active magnetic bearings-chances and limitations. In *IFTOMM Sixth International Conference on Rotor Dynamics, Sydney, Australia*, volume 1, pages 1–14, 2002.
- [34] 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.
- [35] K. Tammi. *Active vibration control of rotor in desktop test environment*. VTT, 2003.
- [36] J. M. Vance. *Rotordynamics of turbomachinery*. John Wiley & Sons, 1988.
- [37] 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.
- [38] T. Yamamoto and Y. Ishida. *Linear and nonlinear rotordynamics: a modern treatment with applications*, volume 11. John Wiley & Sons, 2001.
- [39] 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 [7]):

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 neutral 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

$$\epsilon = -\frac{y}{\mu} \quad (\text{A.1})$$

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

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

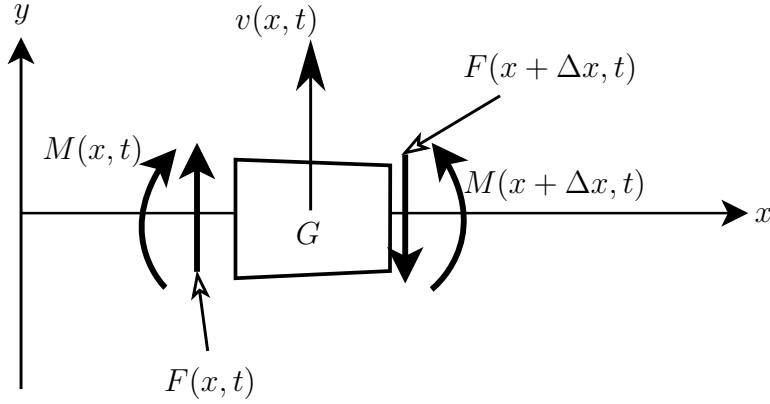


Figure A.1: Free body diagram of a beam section in planar bending.

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:

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

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 known 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

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

and moment equation

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

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

$$\frac{\partial F}{\partial x} = -\rho A \frac{\partial^2 v}{\partial t^2} \quad (\text{A.6})$$

$$\frac{\partial M}{\partial x} - F = 0 \quad (\text{A.7})$$

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

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

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

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

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 [16].

Appendix B

ROTORFEM MATLAB CODE FOR CONSTRUCTING AND ANALYZING FEM MODELS

B.1 Main Object File

```
1 classdef RotorFEM< handle
2 %MODEL Summary of this class goes here
3 % Detailed explanation goes here
4
5 properties
6     M
7     C
8     K
9     F
10    input_file
11    npos
12
13 end
14 properties (Access = private)
15
16 end
17 methods
18     function obj = Model(input_filename)
19         if nargin == 1
20             obj.Assem(input_filename);
21             obj.input_file = input_filename;
22         end
23     end
24 end
25
26 end
```

B.2 Matrix Assembly File

```
1 function Assem(obj, input_file)
2 run(input_file)
3
4 nodes = size(elems,1)+1;
5 npos = zeros(nodes,1);
6 for ii = 1:nodes-1
7     npos(ii+1) = npos(ii) + elems(ii,2);
8 end
9 obj.npos = npos;
10
11 K = zeros(nodes*DOF, nodes*DOF);
```

```

12 C = zeros( nodes*DOF, nodes*DOF) ;
13 M = zeros( nodes*DOF, nodes*DOF) ;
14 F = zeros( nodes*DOF, 1) ;
15 for ii = 1:1:size(elems,1)
16     E = mate(elems(ii,1),2);
17     l = elems(ii,2);
18     do = elems(ii,3);
19     if size(elems(ii,:))<4
20         di=0;
21     else
22         do=elems(ii,4);
23     end
24     I = obj.AInert(do,di);
25     A = obj.Area(do,di);
26     rho = mate(elems(ii,1),3);
27     nuv = mate(elems(ii,1),4);
28     poi = mate(elems(ii,1),5);
29     Id = obj.DiaInert(do,l,rho,di);
30     Ip = obj.PolInert(do,l,rho,di);
31     [kbe, cbe, mbe] = obj.TBeam(E,I,l,A,rho,Id,Ip,nuv,poi,DOF);
32     K = obj.Add(K,kbe,(ii-1)*DOF+1:(ii+1)*DOF);
33     C = obj.Add(C,cbe,(ii-1)*DOF+1:(ii+1)*DOF);
34     M = obj.Add(M,mbe,(ii-1)*DOF+1:(ii+1)*DOF);
35 end
36
37 if exist('disks','var')
38     for ii = 1:1:size(disks,1)
39         do = disks(ii,3);
40         l = disks(ii,2);
41         rho = mate(disks(ii,1),3);
42         a = disks(ii,4);
43         Ki = disks(ii,5);
44         if size(disks(ii,:))<7
45             di=0;
46         else
47             di=disks(ii,7);
48         end
49         md = obj.Mass(do,l,rho,di);
50         Ip = obj.PolInert(do,l,rho,di);
51         Id = obj.DiaInert(do,l,rho,di);
52         [mi,gi,fi] = obj.Disk(md,Id,Ip,a,Ki,DOF);
53         M = obj.Add(M,mi,(disks(ii,6)*DOF-DOF+1:disks(ii,6)*DOF));
54         C = obj.Add(C,gi,(disks(ii,6)*DOF-DOF+1:disks(ii,6)*DOF));
55         F = obj.AddVect(F,fi,(disks(ii,6)*DOF-DOF+1:disks(ii,6)*DOF));
56     end
57 end
58
59 for ii = 1:1:size(bears,1)
60     kx = beartypes(bears(ii,1),2);
61     ky = beartypes(bears(ii,1),3);
62     c = beartypes(bears(ii,1),4);
63     [ki,ci] = obj.Bear(kx,ky,c,DOF);
64     K = obj.Add(K,ki,(bears(ii,2)*DOF-DOF+1:bears(ii,2)*DOF));
65     C = obj.Add(C,ci,(bears(ii,2)*DOF-DOF+1:bears(ii,2)*DOF));
66

```

```

67 end
68 if exist('mags', 'var')
69   for ii = 1:size(mags,1)
70     olddir = pwd;
71     cd(char(magfile(mags(ii,1),2)))
72     Magfn = str2func(char(magfile(mags(ii,1),3)));
73     [ki, ci] = Magfn();
74     cd(olddir)
75     K = obj.Add(K, ki, (mags(ii,2)*DOF-DOF+1:mags(ii,2)*DOF));
76     C = obj.Add(C, ci, (mags(ii,2)*DOF-DOF+1:mags(ii,2)*DOF));
77
78   end
79 end
80 obj.M = M;
81 obj.C = C;
82 obj.K = K;
83 obj.F = F;

```

B.3 Elemental Matrices

```

1 function [ K,D,M ] = TBeam( obj, E,I,l,A,rho,Id,Ip,nuv,poi,DOF )
2 %TBeam Timoshenko beam element
3 % dispLy dispLz angLy angLz disp2y disp2z ang2y ang2z dispLx angLx disp2x ang2x
4 % -----
5 %|                                         | dispLy
6 %|                                         | dispLz
7 %|                                         | angLy
8 %|                                         | angLz
9 %|                                         | disp2y
10 %|                                         | disp2z
11 %|                                         | ang2y
12 %|                                         | ang2z
13 %| ----- | ang2x
14 %% Properties %%
15 k = 6*(1+poi)/(7+6*poi);
16 G = E/2/(1+poi);
17 alph = 12*E*I/k/G/A/l^2;
18 %% Variables %%
19 syms x
20 z = x/l;
21 %% Shape Functions %%
22 N1 = 1-z;
23 N2 = z;
24 Tt1 = (1/(1+alph))*(2*z^3 - 3*z^2 - alph*z + 1 + alph);
25 Tt2 = (1/(1+alph))*(-2*z^3+3*z^2+alph*z);
26 Tr1 = (1/(1+alph))*(z^3 - (2+1/2*alph)*z^2 + (1+1/2*alph)*z);
27 Tr2 = (1/(1+alph))*(z^3 - (1-1/2*alph)*z^2 - 1/2*alph*z);
28 Rt1 = 6/l*(1/(1+alph))*(z^2-z);
29 Rt2 = -Rt1;
30 Rr1 = (1/(1+alph))*(3*z^2 - (4+alph)*z + 1 + alph);
31 Rr2 = (1/(1+alph))*(3*z^2 - (2-alph)*z);
32 %% Transformation Matrices %%

```

```

33 P = [0 0 0 0 0 0;
34     0 0 0 0 0 0;
35     0 -1 0 0 0 0;
36     1 0 0 0 0 0;
37     0 0 0 0 0 0;
38     0 0 0 0 0 0];
39 Di = [12*E*I/alph/l^2          0          0          0          0;
40           0          12*E*I/alph/l^2 0          0          0;
41           0          0          E*I 0          0;
42           0          0          0          E*I 0;
43           0          0          0          0          E*A;
44           0          0          0          0          0          12*E*I/alph/l^2/A*2*I];
45 Mi = diag([rho*A*1,rho*A*1,Id,Id,rho*A*1,Ip])./1;
46 Gi = kron(diag([0,1,0]),[0,Ip;-Ip,0])./1;
47 T = kron(diag([1,1,0]),[0,1;-1,0]);
48 N = [ Tt1 0 0 Tr1 Tt2 0 0 Tr2 0 0 0 0;
49     0 Tt1 Tr1 0 0 Tt2 Tr2 0 0 0 0 0;
50     0 Rt1 Rr1 0 0 Rt2 Rr2 0 0 0 0 0;
51     Rt1 0 0 Rr1 Rt2 0 0 Rr2 0 0 0 0;
52     0 0 0 0 0 0 0 0 N1 0 N2 0;
53     0 0 0 0 0 0 0 0 0 N1 0 N2];
54 %% DOF Modifications%%
55 switch DOF
56 case 6
57     I = diag([1,1,-1,1,1,1]);
58     N = I*N*[I,zeros(6);zeros(6),I]; % Adjust for use of -angy,-angl,2y definitions in shape
59     functions
60 case 4
61     P = P(1:4,1:4);
62     Di = Di(1:4,1:4);
63     Mi = Mi(1:4,1:4);
64     Gi = Gi(1:4,1:4);
65     T = T(1:4,1:4);
66     I = diag([1,1,-1,1]); % Adjust for use of -angy,-angl,2y definitions in shape functions
67     N = I*N(1:4,1:8)*[I,zeros(4);zeros(4),I];
68 case 2
69     P = P(2:3,2:3);
70     Di = Di(2:3,2:3);
71     Mi = Mi(2:3,2:3);
72     Gi = Gi(2:3,2:3);
73     T = T(2:3,2:3);
74     N = N(2:3,[2,3,6,7]);
75 end
76 %% Elemental Integration %%
77 B = diff(N.' ,x) - N.*P;
78 B = B.' ;
79 K_B = int(B.' *Di*B,x,0,1);
80 K_C = int(B.' *T*Di*B,x,0,1);
81 G = int(N.' *Gi*N,x,0,1);
82 M = int(N.' *Mi*N,x,0,1);
83 %% Case of symbolic Integration %%
84 if isa(l,'sym') || isa(E,'sym') || isa(I,'sym') || isa(nuv,'sym')
85 else
86     K_B = double(K_B);
87     K_C = double(K_C);

```

```

87      G = double(G);
88      M = double(M);
89  end
90 K =      K_B + nuv.*1i.*K_C;
91 D = nuv.*K_B +      1i.*G;
92
93 end

```

```

1 function [Md,Gd,Fd] = Disk(obj,md,Id,Ip,a,Ki,DOF)
2 %% Creates Disk nodal stiffness matrix in the form:
3 % disp1x disp1y ang1x ang1y
4 %
5 %| | disp1x
6 %| | disp1y
7 %| | ang1x
8 %| | ang1y
9
10 %% Begin Function
11 Fx = md*a;
12 Fy = md*a;
13 Mx = (Id-Ip)*Ki;
14 My = (Id-Ip)*Ki;
15 Gi=[0,Ip;-Ip,0];
16 switch DOF
17 case 6
18     Md = diag([md,md,Id,Id,md,Ip]);
19     Gd = 1i*kron(diag([0,1,0]),Gi);
20     Fd = [Fx; Fy; Mx; My; 0; 0];
21 case 4
22     Md = diag([md,md,Id,Id]);
23     Gd = 1i*kron(diag([0,1]),Gi);
24     Fd = [Fx; Fy; Mx; My];
25 case 2
26     Md = diag([md,Id]);
27     Gd = 1i*[0,0;0,Ip];
28     Fd = [Fx; My];
29 end
30 % Fx = md*a*(w^2*cos(theta+phi1)-alpha*sin(theta+phi1)); %N, Forcing funtion at disk 1
31 % Fy = md*a*(w^2*sin(theta+phi1)+alpha*cos(theta+phi1)); %N, Forcing funtion at disk 2
32 % Mx = -w^2*(Ip - Id)*Ki*cos(theta+phi2); %N, Forcing funtion at disk 1
33 % My = w^2*(Ip - Id)*Ki*sin(theta+phi2); %N, Forcing funtion at disk 2;
34 end

```

```

1 function [ Kb,Cb ] = Bear(obj,kx,ky,c,DOF)
2 %% Creates bearing nodal stiffness matrix in the form:
3 % disp1z disp1y ang1z disp1x ang1x
4 %
5 %| | disp1y
6 %| | disp1z
7 %| | ang1y
8 %| | ang1z
9 %| | disp1x
10 %| | ang1x
11 %% Begin function
12 switch DOF

```

```

13     case 6
14         Kb = diag([kx,ky,0,0,0,0]);
15         Cb = diag([c*kx,c*ky,0,0,0,0]);
16     case 4
17         Kb = diag([kx,ky,0,0]);
18         Cb = diag([c*kx,c*ky,0,0]);
19     case 2
20         k = mean([kx,ky]);
21         Kb = diag([k,k]);
22         Cb = diag([c*k,c*k]);
23 end
24 end

```

B.4 Plotting Functions

```

1 function RootLocus(Model_obj, Omega, plotmodes, ax, linetp, color)
2 %CAMPBELL Summary of this function goes here
3 % Detailed explanation goes here
4 zci = @(v) find(v(:).*circshift(v(:), [-1 0]) <= 0);
5 if nargin < 6
6     color = [0,0,0];
7     if nargin < 5
8         linetp = ' .';
9     end
10    if nargin < 4
11        figure
12        ax = axes;
13    end
14    if nargin < 3
15        plotmodes = 1:2;
16    end
17    if nargin < 2
18        return
19    end
20
21 end
22
23 for ii = 1:length(Omega)
24     w = Omega(ii)/60*2*pi;
25     Mnew = Model_obj.M;
26     Cnew = real(Model_obj.C) + w.*imag(Model_obj.C);
27     Knew = real(Model_obj.K) + w.*imag(Model_obj.K);
28     Zer = zeros(size(Mnew));
29     ey = eye(size(Mnew));
30     A = [-Mnew^-1*Cnew, -Mnew^-1*Knew;
31           ey, Zer];
32     v1=eig(A);
33 %     AA=[Cnew Mnew;Mnew Zer];
34 %     B=[Knew Zer;Zer -Mnew];
35 %     v1=eig(B,-1i*AA);
36     [~, I] = (sort(abs((v1))));
37     eiv(:,ii) = v1((I));

```

```

38 end
39 % while min(abs(imag(eiv(1,:)))) == 0
40 %     eiv(1:2,:) = [];
41 % end
42 % eiv = eiv(~any(isnan(eiv) | isnan(eiv), 2), :);
43
44 hold on
45 for jj = 1:length(plotmodes)
46
47 % plot(ax,real(eiv(plotmodes(jj)*4-1,:)),(imag(eiv(plotmodes(jj)*4-1,:))),linetp,'Color
48 %',[0.8500,0.3250,0.0980])
49 % plot(ax,real(eiv(plotmodes(jj)*4-3,:)),(imag(eiv(plotmodes(jj)*4-3,:))),linetp,'Color
50 %',[0,0.4470,0.7410])
51 plot3(real(eiv(plotmodes(jj)*4-1,:)),abs(imag(eiv(plotmodes(jj)*4-1,:))),Omega,linetp,'Color',
      color,'MarkerSize',1);%, 'o','Color',[0.8500,0.3250,0.0980], 'MarkerSize',3)
52 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)
53 % plot3(real(eiv(plotmodes(jj)*4-0,:)),imag(eiv(plotmodes(jj)*4-0,:)),Omega,'.', 'Color
54 %',[0,0.4470,0.7410])
55 % plot3(real(eiv(plotmodes(jj)*4-2,:)),imag(eiv(plotmodes(jj)*4-2,:)),Omega,'.', 'Color
56 %',[0,0.4470,0.7410])
57 % r(1,:) = real(eiv(plotmodes(jj)*4-1,:));
58 % r(2,:) = real(eiv(plotmodes(jj)*4-3,:));
59 % ZeroCross1 = zci(r(1,:));
60 % ZeroCross2 = zci(r(2,:));
61 % if isempty(ZeroCross1)
62 % else
63 % strcross1 = [num2str(Omega(ZeroCross1(1)),4) '(RPM)' newline];
64 % text(ax,0,abs(imag(eiv(plotmodes(jj)*4-1,ZeroCross1(1)))),strcross1,'HorizontalAlignment','
65 % right','VerticalAlignment','bottom','FontWeight','bold');
66 % plot(ax,0,abs(imag(eiv(plotmodes(jj)*4-1,ZeroCross1(1)))),'*k');
67 % end
68 % if isempty(ZeroCross2)
69 % else
70 % strcross2 = [num2str(Omega(ZeroCross2(1)),4) '(RPM)' newline];
71 % text(ax,0,abs(imag(eiv(plotmodes(jj)*4-3,ZeroCross2(1)))),strcross2,'HorizontalAlignment','
72 % right','VerticalAlignment','bottom','FontWeight','bold');
73 % plot(ax,0,abs(imag(eiv(plotmodes(jj)*4-3,ZeroCross2(1)))),'*k');
74 % end
75 end
76 hold off
77 ax.YAxisLocation = 'origin';
78 axis([-inf,inf,0,inf])
79 xlabel('Real(s)')
80 ylabel('Imag(s)')
81 end

```

```

1 function Campbell(Model_obj, Omega, plotmodes, ax, linetp )
2 %CAMPBELL Summary of this function goes here
3 % Detailed explanation goes here
4 if nargin < 5
5     if nargin == 2
6         plotmodes = 1:2;
7     elseif nargin < 2
8         return

```

```

9      end
10     linetp = ' .';
11     if nargin < 4
12         ax = [];
13     end
14 end
15 for ii = 1:length(Omega)
16     w = Omega(ii)/60*2*pi;
17     Mnew = Model_obj.M;
18     Cnew = real(Model_obj.C) + w.*imag(Model_obj.C);
19     Knew = real(Model_obj.K) + w.*imag(Model_obj.K);
20     Zer = zeros(size(Mnew));
21     ey = eye(size(Mnew));
22     A = [-Mnew^-1*Cnew, -Mnew^-1*Knew;
23           ey, Zer];
24     v1=eig(A);
25     [~, I] = sort(abs((v1)));
26     eiv(:,ii) = v1(I);
27 % eiv(:,ii)=v1;
28 end
29 if isempty(ax)
30     figure
31     ax = axes;
32 end
33 % while min(abs(imag(eiv(1,:)))) == 0
34 %     eiv(1:2,:) = [];
35 % end
36 % eiv = eiv(~any(isnan(eiv) | isnan(eiv), 2), :);
37
38 hold on
39 for jj = 1:length(plotmodes)
40     plot(ax,Omega,abs(imag(eiv(plotmodes(jj)*4-2,:))/2/pi*60,'.k');%, 'Color',[0,0.4470,0.7410], 'LineWidth',1)
41     plot(ax,Omega,abs(imag(eiv(plotmodes(jj)*4,:))/2/pi*60,'.k');%, 'Color
42 %         [0.8500,0.3250,0.0980], 'LineWidth',1, 'MarkerSize',4)
43 %         plot(ax,Omega,(imag(eiv(plotmodes(jj)*4-3,:))/2/pi*60,linetp,'Color
44 %             ,[0,0.4470,0.7410], 'LineWidth',1)
45 %         plot(ax,Omega,(imag(eiv(plotmodes(jj)*4,:))/2/pi*60,linetp, 'Color
46 %             ,[0.8500,0.3250,0.0980], 'LineWidth',1)
47 %
48 %         plot(ax,Omega,imag(eiv(plotmodes(jj),:))/2/pi*60,linetp,'Color',[0,0.4470,0.7410], 'LineWidth
49 %             ',1)
50
51 end
52 plot(Omega,Omega,'--k','LineWidth',2)
53 % plot(Omega,-Omega,'--k','LineWidth',2)
54 hold off
55 xlabel('Spin speed [RPM]')
56 ylabel('Whirl speed [RPM]')
57 title('Campbell Diagram')
58
59
60 end

```

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

```

2 %STABILITY Summary of this function goes here
3 % Detailed explanation goes here
4 zci = @(v) find(v(:).*circshift(v(:), [-1 0]) <= 0);
5 if nargin < 4
6     linetp = '-';
7     if nargin < 3
8         figure
9         ax = axes;
10    elseif nargin < 2
11        return
12    end
13 end
14
15
16 for ii = 1:length(Omega)
17     w = Omega(ii)/60*2*pi;
18     Mnew = Model_obj.M;
19     Cnew = real(Model_obj.C) + w.*imag(Model_obj.C);
20     Knew = real(Model_obj.K) + w.*imag(Model_obj.K);
21
22     Zer = zeros(size(Mnew));
23     ey = eye(size(Mnew));
24     A = [-Mnew^-1*Cnew, -Mnew^-1*Knew;
25           ey, Zer];
26     v1=eig(A);
27     eiv(:,ii) = sort(v1);
28 end
29 while min(abs(imag(eiv(1,:)))) == 0
30     eiv(1:2,:) = [];
31 end
32 eiv = eiv(~any(isnan(eiv) | isnan(eiv), 2), :);
33 hold on
34 plot(Omega,max(real(eiv)), 'LineWidth', 1, 'LineStyle', linetp);
35 ZeroCross = Omega(zci(max(real(eiv))));
36 if ~isempty(ZeroCross)
37     strcross = [' Threshold: ' num2str(ZeroCross(1), 4) 'RPM'];
38     text(ZeroCross(1), 0, strcross, 'HorizontalAlignment', 'left', 'VerticalAlignment', 'middle',
39          'FontWeight', 'bold', 'Rotation', 90);
40     plot(ZeroCross(1), 0, '*k'); hold off
41 end
42 ax.XAxisLocation = 'origin';
43 xlabel('Speed (RPM)')
44 ylabel('Maximum \Re(s)')
45 title('Stability region')
end

```

```

1 function Damping(Model_obj, Omega, plotmodes, ax, linetp, ch)
2 %CAMPBELL Summary of this function goes here
3 % Detailed explanation goes here
4 zci = @(v) find(v(:).*circshift(v(:), [-1 0]) <= 0);
5
6 if nargin < 6
7     if nargin == 2
8         plotmodes = 1:2;
9     elseif nargin < 2

```

```

10      return
11  end
12  ch = 'dec';
13  if nargin < 5
14      linetp = '-';
15  end
16  if nargin < 4
17      figure
18      ax = axes;
19  end
20 end
21
22 for ii = 1:length(Omega)
23     w = Omega(ii)/60*2*pi;
24     Mnew = Model_obj.M;
25     Cnew = real(Model_obj.C) + w.*imag(Model_obj.C);
26     Knew = real(Model_obj.K) + w.*imag(Model_obj.K);
27
28     Zer = zeros(size(Mnew));
29     ey = eye(size(Mnew));
30     A = [-Mnew^-1*Cnew, -Mnew^-1*Knew;
31           ey, Zer];
32     v1=eig(A);
33     [~, I] = (sort(abs((v1))));
34     eiv(:,ii) = v1((I));
35 end
36
37 while min(abs(imag(eiv(1,:)))) == 0
38     eiv(1:2,:) = [];
39 end
40 eiv = eiv(~any(isnan(eiv) | isnan(eiv), 2), :);
41
42 hold on
43 for jj = 1:length(plotmodes)
44     idf = plotmodes(jj)*4-1; %forward mode index location
45     idb = idf - 2; %backward mode index location
46     zeta = -real(eiv([idb,idf],:))/abs(eiv([idb,idf],:));
47 %     delta = 2*pi.*zeta./sqrt(1-zeta.^2); %Logarithmic Decrement
48 %     delta = -2*pi*real(eiv([idb,idf],:))./imag(eiv([idb,idf],:));
49     plot(Omega,zeta(1,:),linetp,'Color',[0,0.4470,0.7410],'LineWidth',1);
50     plot(Omega,zeta(2,:),linetp,'Color',[0.8500,0.3250,0.0980],'LineWidth',1);
51     ZeroCross1 = Omega(zci(zeta(1,:)));
52     ZeroCross2 = Omega(zci(zeta(2,:)));
53     if isempty(ZeroCross1)
54         else
55             strcross1 = [num2str(ZeroCross1(1),4) newline];
56             text(ZeroCross1(1),0,strcross1,'HorizontalAlignment','left','VerticalAlignment','bottom',...
57                  'FontWeight','bold');
58             plot(ZeroCross1(1),0,'*k');
59         end
60         if isempty(ZeroCross2)
61             else
62                 strcross2 = [ num2str(ZeroCross2(1),4) newline];
63                 text(ZeroCross2(1),0,strcross2,'HorizontalAlignment','left','VerticalAlignment','bottom',...
64                     'FontWeight','bold');

```

```

63 plot(ZeroCross2(1),0,'*k');
64 end
65
66 end
67 hold off
68 ax.XAxisLocation = 'origin';
69 xlabel('Speed (RPM)')
70 ylabel('\zeta')
71 title('Damping Characteristics')
72
73 end

```

```

1 function Shape(Model_obj, Omega, plotmodes, ax, linetp )
2 %CAMPBELL Summary of this function goes here
3 % Detailed explanation goes here
4 if nargin < 5
5     if nargin < 2
6         return
7     end
8     linetp = '-';
9     if nargin < 4
10        figure
11        ax = axes;
12    end
13 end
14
15 syms x;
16 l=1;
17 z=x/l;
18 alph=.25;
19 Tt1 = (1/(1+alph))*(2*z^3 - 3*z^2 - alph*z + 1 + alph);
20 Tt2 = (1/(1+alph))*(-2*z^3+3*z^2+alph*z);
21 Tr1 = (1/(1+alph))*(z^3 - (2+1/2*alph)*z^2 + (1+1/2*alph)*z);
22 Tr2 = (1/(1+alph))*(z^3 - (1-1/2*alph)*z^2 - 1/2*alph*z);
23 cpsi = [Tt1 0 0 Tr1 Tt2 0 0 Tr2;
24 0 Tt1 -Tr1 0 0 Tt2 -Tr2 0];
25 % psi = [1 - 3*(x/1)^2 + 2*(x/1)^3, x - 2*1*(x/1)^2 + 1*(x/1)^3, 3*(x/1)^2 - 2*(x/1)^3, -1*(x/1)^2
26 % + 1*(x/1)^3];
27 % cpsi = [psi(1), 0, 0, psi(2), psi(3), 0, 0, psi(4);
28 % 0, psi(1), -psi(2), 0, 0, psi(3), -psi(4), 0];
29 ipts = 4;
30 d = (0:ipts-1)/ipts;
31 for ii = 1:l:length(d)
32     Nx(ii,:) = subs(cpsi(1,:),x,d(ii));
33     Ny(ii,:) = subs(cpsi(2,:),x,d(ii));
34 end
35 Nx = double(Nx);
36 Ny = double(Ny);
37 w = Omega/60*2*pi;
38 Mnew = Model_obj.M;
39 Cnew = real(Model_obj.C) + w.*imag(Model_obj.C);
40 Knew = real(Model_obj.K) + w.*imag(Model_obj.K);
41 nM = length(Mnew);
42 % Zer = zeros(nM);

```

```

43 % AA=[Cnew Mnew;Mnew Zer];
44 % B=[Knew Zer;Zer -Mnew];
45 Zer = zeros( size(Mnew));
46 ey = eye( size(Mnew));
47 A = [-Mnew^-1*Cnew, -Mnew^-1*Knew;
48 ey , Zer];
49
50 [V,D] = eig(A, 'vector');
51 % ind=find( (imag(D) > 0) & (abs(D)>1e-3) );      % take positive frequencies only
52 % l1=D(ind);
53 % v1=V(nM+1:end ,ind );
54 l1=D;
55 v1=V(nM+1:end ,: );
56 % v1=V(1:nM,: );
57 [~, I] = sort(abs((l1)));
58 l2 = D(I);
59 v2 = V(:, I);
60 v = v2(:, plotmodes);
61 v=v/norm(v);
62 for ii = 1:1:length(Model_obj.M)/4-1
63 Li = Model_obj.npos(ii+1)-Model_obj.npos(ii);
64 Nxnew = Nx*diag([1 1 1 Li 1 1 Li]);
65 Nynew = Ny*diag([1 1 Li 1 1 1 Li 1]);
66 pos(ipts*(ii-1)+1:ipts*(ii-1)+ipts+1) = linspace(Model_obj.npos(ii),Model_obj.npos(ii+1),ipts
+1);
67 pos(end)=[];
68 shapex(ipts*(ii-1)+1:ipts*(ii-1)+ipts) = Nxnew*v(4*(ii-1)+1:4*(ii-1)+8);
69 shapey(ipts*(ii-1)+1:ipts*(ii-1)+ipts) = Nynew*v(4*(ii-1)+1:4*(ii-1)+8);
70 end
71 pos = [pos, Model_obj.npos(end)];
72 shapex = [shapex, v(nM-3)];
73 shapey = [shapey, v(nM-2)];
74 plot(ax,[pos;pos]',real([shapex; shapey])')
75 hold on
76 for ii = 1:length(Model_obj.npos)
77 plot([Model_obj.npos(ii);Model_obj.npos(ii)]',real([shapex(ipts*(ii-1)+1); shapey(ipts*(ii-1)
+1)])','.k','MarkerSize',8)
78 end
79 hold off
80 figure
81 cpts = 20;
82 theta = linspace(0,2*pi,cpts);
83 for ii = 1:length(shapex)
84 xx(:,ii)=real(shapex(ii))*cos(theta) - imag(shapex(ii))*sin(theta);
85 y(:,ii)=real(shapey(ii))*cos(theta) - imag(shapey(ii))*sin(theta);
86 end
87 N = repmat(pos,cpts,1);
88 hold on
89 plot3(real(shapex),pos,real(shapey),'-k')
90
91 plot3(xx,N,y,'Color',[0,0.4470,0.7410])
92
93 for ii = 1:length(Model_obj.npos)
94 plot3(real(shapex(ipts*(ii-1)+1)),Model_obj.npos(ii),real(shapey(ipts*(ii-1)+1)),'.k',
'MarkerSize',8)

```

```

95 end
96
97 hold off
98 mx = max([abs(shapeX), abs(shapeY)]);
99 axis([-mx, mx, -inf, inf, -mx, mx]);
100 disp(['Damping: ' num2str(real(12(plotmodes))) ' Frequency: ' num2str(imag(12(plotmodes))/2/pi
101 *60) '(RPM) .']);
102 ax=gca;
103 ax.XDir='reverse';
104 ax.Title.String={[['\Re(s): ' num2str(real(12(plotmodes)),2) ', \Im(s): ' num2str(imag(12(
105 plotmodes))/2/pi*60,4) '[RPM]]};{['\Omega: ' num2str(Omega,4) '[RPM]']};
106 %
107 % [V, D] = eig(Knew, Mnew, 'vector');
108 % D = sqrt(D)*60/2/pi;
109 % [D, ind] = sort(abs(D));
110 % D(plotmodes)
111 % V = V(:,ind);
112 % v2 = (V(:,plotmodes));
113 % v3 = reshape(v2,4,[]);
114 % v4 = bsxfun(@times, v3, 1./sqrt(sum(v3.^2, 2)));
115 % v2 = reshape(v4,1,[]);
116 % for ii = 1:length(Model_obj.M)/4-1
117 %     shape(8*(ii-1)+1:8*(ii-1)+8) = N*v2(4*(ii-1)+1:4*(ii-1)+8);
118 % end
119 % shape = [shape, v2(end-3)];
120 % plot(real(shape))
121 %
122 % if isempty(ax)
123 %     figure
124 %     ax = axes;
125 %
126 % hold on
127 % for jj = 1:length(plotmodes)
128 %     pv(1,:) = real(eigect(speed,[4.*(1:14)-3],plotmodes(jj)));
129 %
130 % plot(pv)
131 % end
132 % hold off
133
134
135 end

```

```

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

```

```

11      ax = axes( fi );
12      if nargin < 4
13          plottype = 'Bode';
14      end
15      if strcmp( plottype , 'Bode' ) == 1
16          ax(1)=subplot(2,1,1,ax);
17          ax(2)=subplot(2,1,2);
18      end
19  end
20
21 end
22
23 for ii = 1:1:length(Omega)
24     w = Omega(ii)/60*2*pi;
25     Fnew = w.^2.* (Model_obj.F);
26     Mnew = Model_obj.M;
27     Cnew = real(Model_obj.C) + w.*imag(Model_obj.C);
28     Knew = real(Model_obj.K) + w.*imag(Model_obj.K);
29     n=length(Mnew);
30     I=kron(eye(n/2),diag([1,1i]));
31     Z = I*(Knew + 1i*w.*Cnew - w.^2.*Mnew);
32     X(ii,:)=Z^-1*Fnew;
33 end
34 switch plottype
35     case 'FreqResponse'
36         hold on
37         for jj = 1:1:length(node)
38             ax=fi.Children;
39             plot(ax, Omega,(abs(X(:,node(jj)*4-3))),linetp)
40             plot(ax, Omega,(abs(X(:,node(jj)*4-2))),linetp)
41             ax.YScale = 'log';
42         end
43         hold off
44         ax.XLabel.String='Spin Speed \Omega[RPM]';
45         ax.YLabel.String='Amplitude [m]';
46     case 'Bode'
47         hold on
48         for jj = 1:1:length(node)
49             subplot(211); hold on
50             ax(1)=fi.Children(1);
51 %             plot(Omega,abs(X(:,node(jj)*4-3)),linetp)
52 %             plot(Omega,abs(X(:,node(jj)*4-2)),linetp)
53             plot(ax(1),Omega,abs(39370.1*X(:,node(jj)*4-3)),linetp)
54             plot(ax(1),Omega,abs(39370.1*X(:,node(jj)*4-2)),linetp)
55             ax(1).YScale = 'log';
56             ax(1).XLabel.String='Spin Speed \Omega[RPM]';
57 %             ax(1).YLabel.String='Amplitude [m]';
58             ax(1).YLabel.String='Amplitude [mils]';
59             subplot(212); hold on
60             ax(2)=fi.Children(2);
61 %             plot(Omega,unwrap(mod(angle(X(:,node(jj)*4-3)),2*pi)),linetp)
62 %             plot(Omega,unwrap(mod(angle(X(:,node(jj)*4-2)),2*pi)),linetp)
63             plot(ax(2),Omega,180/pi*unwrap(mod(angle(X(:,node(jj)*4-3)),2*pi)),linetp)
64             plot(ax(2),Omega,180/pi*unwrap(mod(angle(X(:,node(jj)*4-2)),2*pi)),linetp)
65             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

```
1 classdef RotorData < handle
2 properties
3     X %Horizontal Signal
4     Y %Vertical Signal
5     ref %Keyphasor Signal
6     Fs %Sampling Rate (Samp/Sec)
7     Fres %Frequency Resolution
8     n %filter synchronous multiplier: 1X, 2X, 3X, ... nX the running speed
9 end
10 properties (SetAccess = private)
11     Zf %Total Amplitude
12     Phase %Phase
13     Amp %Property that holds X and Y amplitudes
14     Speed %RPM
15 end
16 properties (Access = private)
17     refchop %%%
18     xchop %% Organizes the data arrays into matrices based on windows
19     ychop %%%
20 end
21 properties (Dependent, Access = private)
22     NW %Number of windows
23     freq %Frequency
24     nspw %Number of samples per window
25
26 end
27 methods
28     function obj = RotorData(x, y, r, f, fres)%Fills properties with data
29         if nargin == 5
30             obj.X = x;
31             obj.Y = y;
32             obj.ref = r;
33             obj.Fs = f;
34             obj.Fres = fres;
35         end
36     end
37     function set.X(obj, val)
38         obj.X = val;
39     end
40     function set.Y(obj, val)
41         obj.Y = val;
42     end
43     function set.ref(obj, val)
44         obj.ref = val;
45     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(
49             obj.Fres)
50             else
51                 obj.update;
52             end
53     end
54     function set.Fres(obj, val)
55         obj.Fres = val;
56         if isempty(obj.X) || isempty(obj.Y) || isempty(obj.ref) || isempty(obj.Fs) || isempty(
57             obj.Fres)
58             else
59                 obj.update;
60             end
61     end
62     function set.n(obj, val)
63         obj.n = val;
64         if isempty(obj.X) || isempty(obj.Y) || isempty(obj.ref) || isempty(obj.Fs) || isempty(
65             obj.Fres)
66             else
67                 obj.update;
68             end
69     end
70     function nspw = get.nspw(obj)
71         nspw = ceil(obj.Fs/obj.Fres);
72     end
73     function freq = get.freq(obj)
74         df = obj.Fs/obj.nspw;
75         f = df*(0:obj.nspw-1);
76         Q = ceil((obj.nspw+1)/2); % M/2+1 for M even
77         fQ = df*(Q-1);
78         freq = f-fQ;
79     end
80     function NW = get.NW(obj)
81         NW = floor(length(obj.X)/obj.nspw); % Rounds number of windows down to closest integer
82     end
83     function filter(obj)
84         disp('filtering')
85         for i = 1:1:obj.NW
86             Fpass1 = obj.Speed(i)*obj.n/60-5; %Upper frequency of bandpass filter
87             Fpass2 = obj.Speed(i)*obj.n/60+5; %Lower frequency of bandpass filter
88             if i == 1
89                 h = fdesign.bandpass('N,Fp1,Fp2,Ast1,Ap,Ast2', 10, Fpass1, Fpass2, 50, .1,
90                     50, obj.Fs);
91                 Hd(i) = design(h, 'ellip');
92                 Hd(i).persistentmemory = true;
93                 obj.xchop(:,i) = filter(Hd(i),obj.xchop(:,i)); %Apply filter to data
94                 xf = Hd(i).states;
95                 reset(Hd(i));
96                 obj.ychop(:,i) = filter(Hd(i),obj.ychop(:,i)); %|||
97                 yf = Hd(i).states;
98             else
99                 h = fdesign.bandpass('N,Fp1,Fp2,Ast1,Ap,Ast2', 10, Fpass1, Fpass2, 50, .1,
100                     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
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
127     min and max from each column
128     obj.Amp.YAmp = max(obj.ychop) - min(obj.ychop); %Calculates Amplitude by subtracting
129     min and max from each column
130     Zwin = hanning(obj.nspw, 'Periodic');
131     Z = obj.xchop + 1i*obj.ychop;
132     for i = 1:1:obj.NW
133         Z(:,i) = Zwin.*Z(:,i);
134     end
135     obj.Zf = 2*abs(fft(Z)/obj.nspw);
136     obj.Zf = fftshift(obj.Zf',2); %Shifts graph to center on the x axis
137
138     obj.Phase.PhaseX = zeros(obj.NW,1);
139     obj.Phase.PhaseY = zeros(obj.NW,1);
140     distance = .5*obj.Fs/100;
141     threshold = .5*(max(obj.refchop(:,ceil(end/2))) - mean(obj.refchop(:,ceil(end/2))));%
142     for i = 1:1:obj.NW
143         [~, locsref] = findpeaks(obj.refchop(:,i),'MinPeakHeight',threshold,%
144             'MinPeakDistance',distance);
145         [~, locsX] = findpeaks(obj.xchop(:,i),'MinPeakProminence',1,'MinPeakDistance',%
146             distance);
147         [~, locsY] = findpeaks(obj.ychop(:,i),'MinPeakProminence',1,'MinPeakDistance',%
148             distance);
149         if length(locsref) < 2 || length(locsX) < 2 || length(locsY) < 2
150             obj.Phase.PhaseX(i) = NaN;

```

```

146     obj.Phase.PhaseY(i) = NaN;
147
148     else
149         if length(locsX) ~= length(locsY)
150             if length(locsX) < length(locsY)
151                 if abs(locsX(1) - locsY(1)) < abs(locsX(end) - locsY(end))
152                     locsY(end) = [];
153                 else
154                     locsY(1) = [];
155                 end
156             else
157                 if abs(locsX(1) - locsY(1)) < abs(locsX(end) - locsY(end))
158                     locsX(end) = [];
159                 else
160                     locsX(1) = [];
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                         );
175             phy(j) = mod((locsY(j) - locsref(j))/(locsref(j+1) - locsref(j))*2*pi,2*pi
176                         );
177         end
178         obj.Phase.PhaseX(i) = 180/pi*mod(mean(phx(:)),2*pi);
179         obj.Phase.PhaseY(i) = 180/pi*mod(mean(phy(:)),2*pi);
180         clear locsref locsX locsY j
181     end
182 end
183
184 function acqListener(obj, src, event)
185     obj.newData = [event.TimeStamps, event.Data]';
186     obj.ref = [obj.ref, event.Data(1,:)];
187     obj.X = [obj.X, event.Data(2,:)];
188     obj.Y = [obj.Y, event.Data(3,:)];
189     obj.update;
190
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

```

```
199 | end
```

C.2 Plotting Functions

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

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

```
1
2 function orbit3(obj)
3 N = length(obj.X(:));
4 F = (obj.Speed(end) - obj.Speed(1))/N*(0:N-1) + obj.Speed(1);
5 s = fix(sqrt(N));
6 Xlin = obj.X(1:s^2);
7 Ylin = obj.Y(1:s^2);
8 w = F(1:s^2);
9 x = reshape(Xlin,s,s);
10 y = reshape(Ylin,s,s);
11 o = reshape(w,s,s);
12 C = sqrt(x.^2+y.^2);
13 h = mesh(o,x,y,C);
14 axis([1000 2000 -inf inf -inf inf]);
15 xlabel('Running Speed (RPM)');
16 ylabel('Horizontal Amplitude (mils)');
17 zlabel('Vertical Amplitude (mils)');
18 h.FaceColor = 'none';
19 h.MeshStyle = 'column';
20 end
```

```
1 function orbitAnimation(obj)
2
3 distance = .5*obj.Fs/100;
```

```

4 threshold = .5*(max(obj.refchop(:,ceil(end/2))) - mean(obj.refchop(:,ceil(end/2))));  

5 [~, loc] = findpeaks(obj.ref, 'MinPeakHeight', threshold, 'MinPeakDistance', distance);  

6 pksx=obj.X(loc);  

7 pksy=obj.Y(loc);  

8 Fsh = obj.Fs;  

9 maxlm = max(obj.X);  

10 speedarray = (obj.Speed(end)-obj.Speed(1))/length(pksy)*(0:length(pksy)-1)+obj.Speed(1);  

11 for i = 1:1:length(pksx)-1;  

12 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)  

13 text = [ ' Running Speed: ', num2str(fix(speedarray(i)),4), ' (RPM) '];  

14 uicontrol('Style', 'text', ...  

    'String', text, ...  

    'Units', 'normalized', ...  

    'Position', [.25 .9 0.5 .05]);  

15 axis([-maxlm maxlm -maxlm maxlm]);  

16 axis square  

17 xlabel('Horizontal Amplitude (mils)');  

18 ylabel('Vertical Amplitude (mils)');  

19 % pause(1/Fsh)  

20 pause  

21  

22 end  

23 end

```