Fall 2017

Chatter Simulation and Detection in CNC Milling

Jonathan Steven Shepard
University of New Hampshire, Durham

CHATTER SIMULATION AND DETECTION IN CNC MILLING

BY

JONATHAN SHEPARD

BS Mechanical Engineering, University of New Hampshire, 2014

THESIS

Submitted to the University of New Hampshire

in Partial Fulfillment of

the Requirements for the Degree of

Master of Science

in

Mechanical Engineering

September, 2017

This thesis has been examined and approved in partial fulfillment of the requirements for the degree of Master of Science in Mechanical Engineering by:

Thesis Director, Dr. Barry K. Fussell, Professor of Mechanical Engineering

Dr. Todd Stuart Gross, Professor of Mechanical Engineering

Dr. May-Win L. Thein, Associate Professor of Mechanical Engineering

On 5/19/2017

Original approval signatures are on file with the University of New Hampshire Graduate School.

Dedication

To my mom and dad.

Table of Contents

List of Figures

List of Tables

Abstract

CHATTER SIMULATION AND DETECTION IN CNC MILLING

by

Jonathan Shepard

University of New Hampshire, September, 2017

A key feature to maintaining a safe and efficient machining operation is the avoidance of unstable vibrations of the cutting tool, commonly called chatter. This thesis explores chatter simulation and a new chatter detection algorithm developed for use in a CNC milling machine. This algorithm, based on a once per revolution sampling method, samples forces experienced during the milling process, and calculates the variance of the differences between consecutive samples. As a selected level of variance, chatter is assumed. An existing milling simulation program is modified to produce a stability lobe diagram for a given CNC machine, workpiece material, and cutting tool. Stable, unstable, and marginally stable cuts are indicated as a function of spindle speed and axial depth of cut.

The chatter detection algorithm is initially verified by simulation and then added to the simulation to auto-generate stability lobe diagrams. A collection of experimental aluminum cuts is run to collect force data that can be analyzed by the detection algorithm and compared to simulation results. Experimental cut stability is determined by observation of the noise, force, and cut surface. Comparing algorithm results to observed results shows the effectiveness of the algorithm in distinguishing between stable and unstable cuts. However, further testing is needed, particularly in determining the variance threshold of the algorithm.

The simulation and algorithm are also used to explore the effect of system parameters; specifically, spring constant (k) and damping ratio (ζ). This exploration shows there being a strong connection between the maximum attainable stable axial depth and the parameters.

Chapter 1: Introduction

1.1: Introduction

Machining is one of the most used processes in manufacturing across all industries. Two of the most common forms are turning, where a stationary cutting tool is used to shape a spinning workpiece, and milling, where a spinning cutting tool is used to shape a workpiece mounted to a table. To achieve better accuracy and repeatability, CNC (Computer Numerical Control) machining was developed. This was a major step for its time, but there have been few improvements over the years to make the system smarter. There is a need for automated process monitoring and process improvement to make CNC machining both safe and efficient.

Automated process monitoring and improvement requires methods and algorithms to interpret sensor signals, and determine what actions, if any, should be taken. This process is illustrated in Figure 1, where the CNC controller communicates with the CNC machine to maintain an optimum process. The real-time controller uses sensor information it receives from the CNC machine and part G-code information (pre-processing) to apply wear, force control, and chatter algorithms contained within the controller.

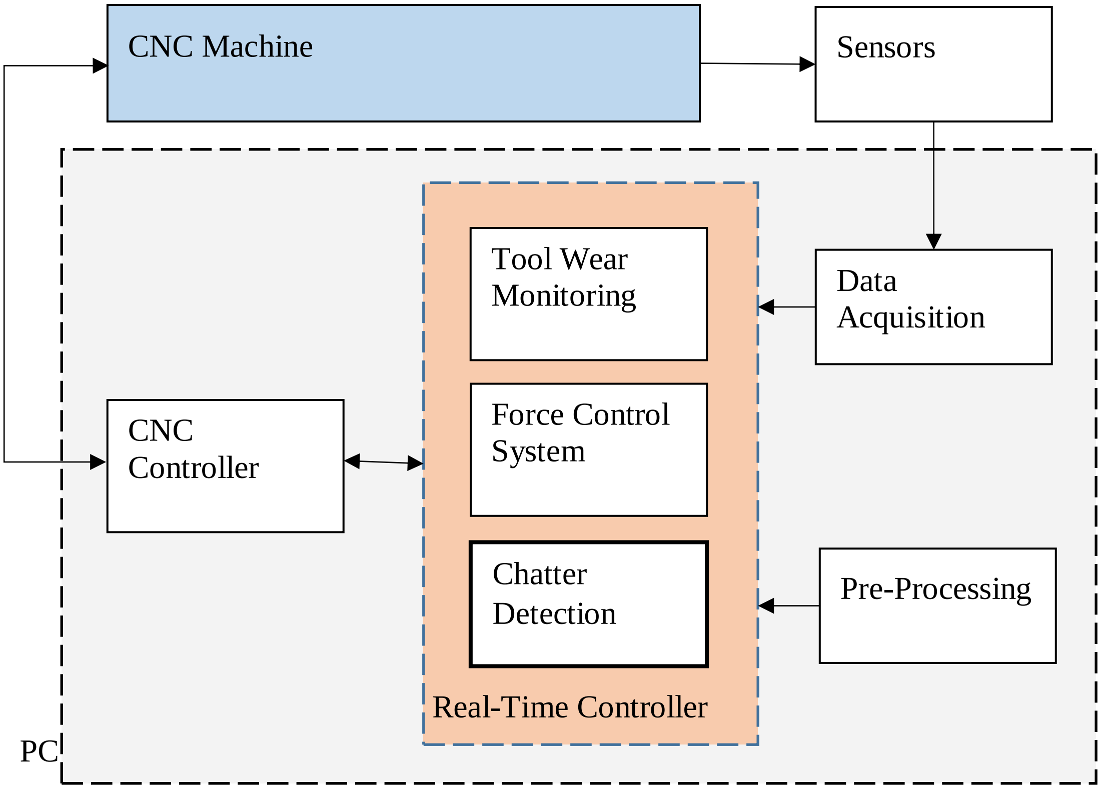


Figure 1: Real-time monitoring and control block diagram.

Previous research has explored tool wear monitoring and force control algorithms [1], which are applied in the real-time controller. The goal of this research is to simulate CNC machining chatter and develop a chatter detection algorithm. This would form the core of an eventual control system that can monitor and adjust for chatter, while working in conjunction with the previously developed controls.

A central feature of safe milling is chatter free cutting. Chatter is an unstable tool vibration during turning and milling operations that results in a rough surface on the workpiece, an example of which is shown in Figure 2. The greatest risk from chatter is the large cutting forces that can occur near the natural frequency of the machine, potentially causing part surface damage and tool breakage. If chatter is allowed to build for an extended time, there is the potential for damage to not only the workpiece and tool, but to the machine as well.

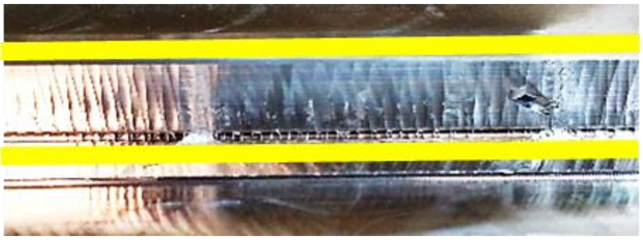


Figure 2: Workpiece with a rough surface caused by chatter. Surface outlined by yellow lines.

Several different approaches of chatter detection have been explored using various sensors such as microphones [2] [3], plate dynamometers [4] [3], and various displacement sensors [3]. All the sensors are used to collect online data, which is analyzed by a data processing method, such as once per revolution sampling [2], Fast Fourier Transform (FFT) analysis [4] [3], and Power Spectral Density (PSD) analysis. The major disadvantage of the microphone is that it is susceptible to ambient noise, while the plate dynamometer is too expensive for industrial use, and the displacement sensors are easily disrupted by chips and coolant expelled during milling. Most of these data processing methods flag chatter by looking for differences between a collected data set and a known stable case. Making comparisons can be a good way to detect chatter, but it requires data collection for new cutting conditions that can slow down detection time, delaying preventative actions.

Figure 3 shows the Fadal CNC milling machine used in this research. The machine contains a 2-axis moving table, along with a vertically moving spindle. The CNC controller is a Windows based, real time operating system, and is programmed using G and M codes. It has an open architecture to allow for the integration of sensors and controllers. One of the main sensors used is a Kistler plate dynamometer, shown in right side of Figure 3. The machine also contains sensors to protect it from current overloads in the feed drive and spindle motor. The machine can handle a large variety of cutting tools, but for this research, only flat end insert cutters are utilized.

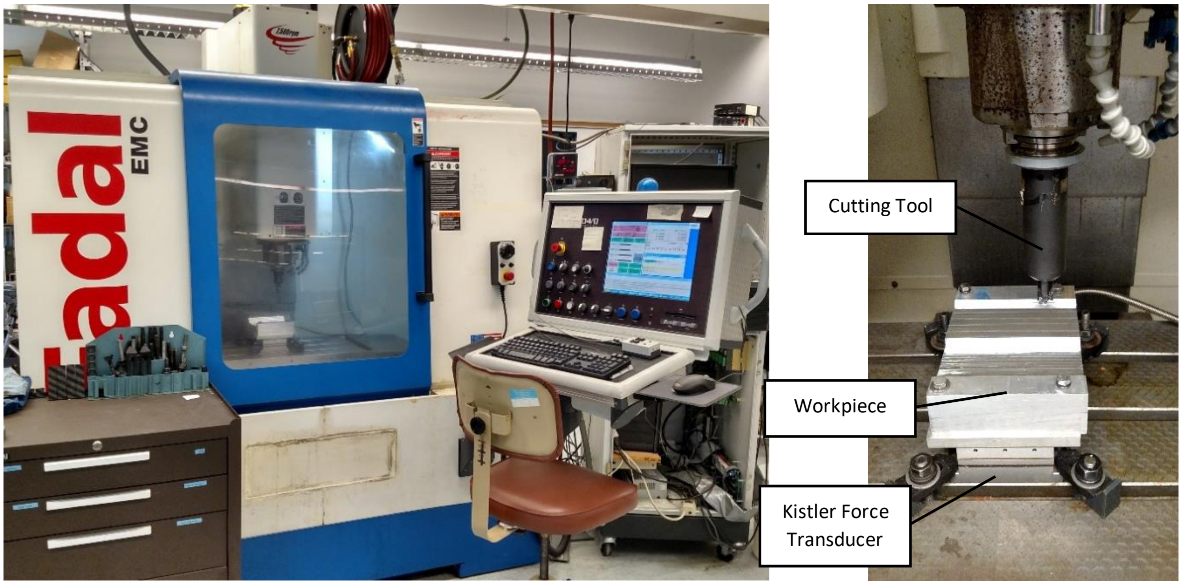


Figure 3: Fadal CNC milling machine. Overall setup (left) and close up of spindle and work piece mounted on Kistler force dynamometer (right).

The purpose of this research is to develop a chatter detection algorithm for use in a CNC milling machine. To efficiently test the effectiveness of the developed algorithm, a MATLAB based simulation is utilized. Experimental cuts are completed for further verification of both the algorithm and simulation results.

1.2: Thesis Overview

This thesis presents the work completed toward chatter simulation and a reliable, cost effective chatter detection algorithm. Chapter 2 discusses the cutting force model, along with the theoretical description of chatter. A linearized, coefficient based model, as described in Altintas [5], is used to calculate cutting forces. The description of chatter is from Schmitz and Smith [6], and is based on the coefficient based cutting model.

Chapter 3 describes the milling simulation used in the research. It starts with an overview of the three portions of the simulation: initialization, force model calculations, and system dynamics calculations. A complete description of the dynamic equations is presented. This chapter also describes the methods used to obtain the system parameters.

Chapter 4 describes previous work in chatter detection, with a focus on the method that uses once per revolution sampling. It continues to a description of the new algorithm developed in this research, and the simulation results used to evaluate the algorithm.

Chapter 5 presents the experimental testing used to validate the detection algorithm. The results from the experiments are presented, evaluated, and compared to simulations.

Chapter 6 examines the effect of CNC machine system parameters on chatter. Several sets of results are presented, each with the spring constant (k) or damping ratio (ζ) shifted up or down to explore their effect on system chatter.

Chapter 7 wraps up the thesis with conclusions and future work. This is followed by the appendix.

Chapter 2: Theory/Background

2.1: Introduction

This chapter introduces the milling process model consisting of a force model, compliance model, and tool deflection feedback that affect the behavior of the system. Under certain cutting conditions the system can become unstable, resulting in machine chatter.

Milling, like many complicated physical processes, can be explored with mathematical models of the process and a computer simulation package that can solve the resulting coupled equations that are often nonlinear. The input to the milling process is the desired chip thickness which is dependent on the feed-rate and spindle speed (Ω). The output is the displacement of the tool which can affect part dimensions and lead to chatter.

The milling process is composed of a force model, a tool compliance model, and the instantaneous and regenerative (previous) deflection feedbacks. These parts are illustrated in the generic block diagram shown in Figure 4. The milling force cutting model and tool compliance model are described in detail in the following sections.

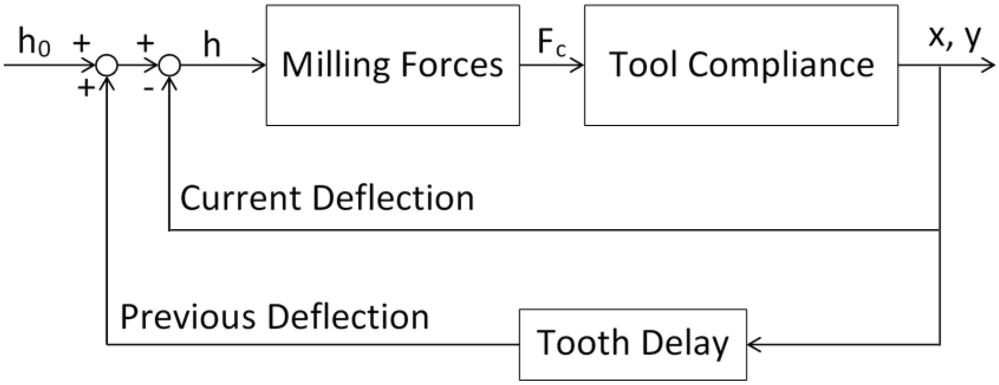


Figure 4: General block diagram for milling process.

The regenerative feedback is positive in the block diagram, and this is significant in chatter development, which is an unstable response of the system to a given chip thickness input and cut geometry. This feedback is positive because the surface left by a negative instantaneous deflection of the previous tooth leaves material that must be removed by the next tooth. This positive feedback can lead to system instability under certain cutting conditions.

2.2: Cutting Force Model

Milling cutting forces can be accurately predicted by the coefficient based mechanistic force model described below [5]. This model has been tested extensively, and shown to have good accuracy and repeatability when the model is calibrated correctly [1]. The tangential and radial forces are shown in Figure 5 and can be written as:

$$F_{tangential}\ [\mathrm{N}] = F_t = K_{tc} a h(\phi) + K_{te} a \tag{1}$$

$$F_{radial}\ [\mathrm{N}] = F_r = K_{rc} a h(\phi) + K_{re} a \tag{2}$$

where *a* is the axial depth of cut [mm], K_{tc} and K_{rc} are the tangential and radial cutting coefficients [N/mm^2], K_{te} and K_{re} are the tangential and radial edge coefficients [N/mm], and ϕ is the angular location of a tooth edge in the cut [deg]. The chip thickness (*h*) [mm], with no tool deflections is:

$$h(\phi) = f_t \sin(\phi) \tag{3}$$

where f_t is the feed per tooth [mm/tooth]. The feed per tooth is proportional to the slide feed-rate and inversely proportional to the spindle speed. The above equations are for a zero-helix cutter, however a cutter with a helix angle can be sliced into numerous disks, and the force on each disk can be summed. This is necessary because the cut geometry on each disk is different because of the helix angle.

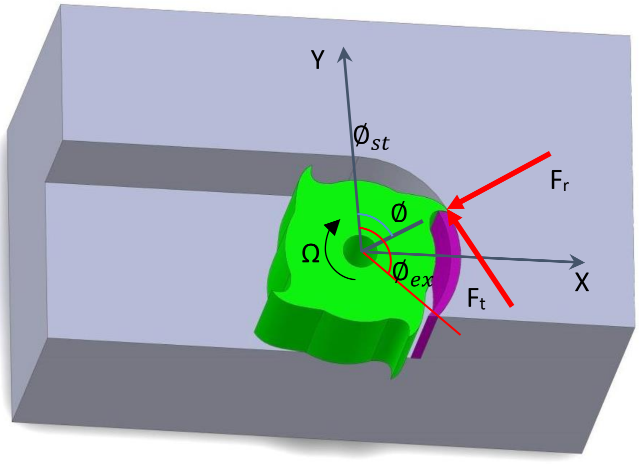


Figure 5: Cutting model forces and coordinates [1]. The figure is assuming up milling, with feed direction in x-direction.

The radial and tangential cutting forces can be resolved into the feed direction (x) and normal direction (y) of the cutter. This formulation was generated for a flat end cutter, with the z-direction force neglected since it is usually very small. The transformations are:

$$F_x\ [\mathrm{N}] = -F_t cos\phi - F_r sin\phi \tag{4}$$

$$F_y\ [\mathrm{N}] = F_t sin\phi - F_r cos\phi \tag{5}$$

The total force experienced by a tool with a helix angle is obtained by summing the forces on each axial slice, as well as each tooth:

$$F_{x\_tool} = \sum_{i=1}^{n} \sum_{j=1}^{m} F_x\,(i,j) \tag{6}$$

$$F_{y\_tool} = \sum_{i=1}^{n} \sum_{j=1}^{m} F_y(i,j) \tag{7}$$

where n is the number of teeth on the tool, and m is the number of axial slices.

This model has been used in previous research, and shown to be reliable. Work done by Eren [7] quantified this reliability, showing there is a maximum error of 14% in peak force prediction for a given cut geometry with recently calibrated coefficients. Nouri [1] shows the accuracy of the simulation's predictions to experimental cuts in Figure 6.

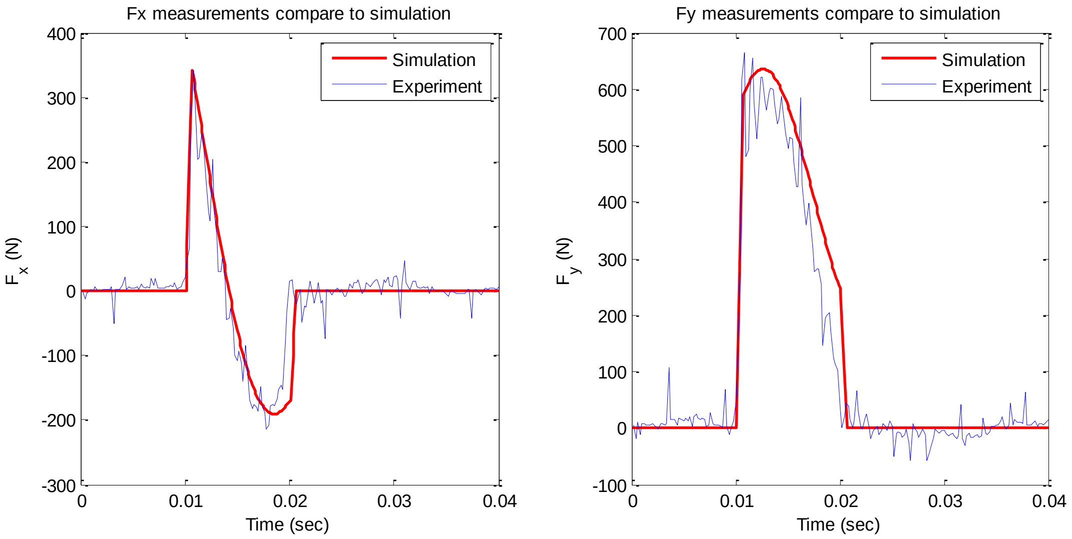


Figure 6: Simulation validation - down milling cut on Steel 1018 with a one tooth 0.75 in diameter cutter at 1500 rpm spindle speed. The axial and radial depth are 0.1 and 0.375 in (.25 and .95 mm). Feedrate is 5.44 in/min (0.14 m/min). (Nouri [1])

2.3: Milling Process Model

The system's dynamics are modeled as a two-dimensional spring-mass-damper system, the diagram for which is shown in Figure 7. Like the force calculations, z-direction motion is considered negligible due to the stiffness of the tool in that direction. For this configuration, the differential equations for this system are:

$$m\ddot{x} + c_x\dot{x} + k_x x = F_x \quad (8)$$

$$m\ddot{y} + c_y\dot{y} + k_y y = F_y \quad (9)$$

where m is the effective mass [kg] of the tool/holder/spindle, c is the damping coefficient [Ns/m] of the tool/workpiece interface/spindle, k is the spring coefficient [N/m] of the tool/holder/spindle, F is the experienced force [N], and *x* and *y* are the displacements [m] in their respective directions. The equations could be solved for x and y by using a simple integration routine.

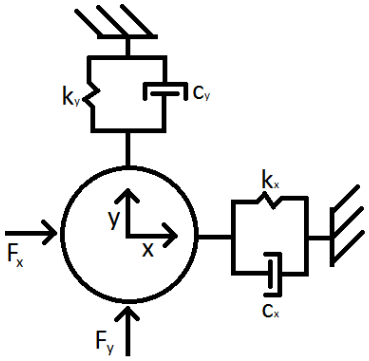


Figure 7: Single mode spring-mass-damper system in x and y.

2.4: Chatter Theory

Chatter can occur during both turning and milling operations. The vibrations are initiated by the system dynamics in response to the regenerative and instantaneous deflection feedbacks. These vibrations are amplified when the wavy surface created on the workpiece surface (regenerative) is out of phase with the instantaneous deflection.

Referring to Figure 4, it can be observed that the regenerative deflection provides a positive feedback to the system loop. Based on classic control theory, positive feedback is undesired since it is a major source of instability in the system. An image of milling force during chatter is shown in Figure 8. Note how the force peaks are large, and the forces widely vary between consecutive tool revolutions.

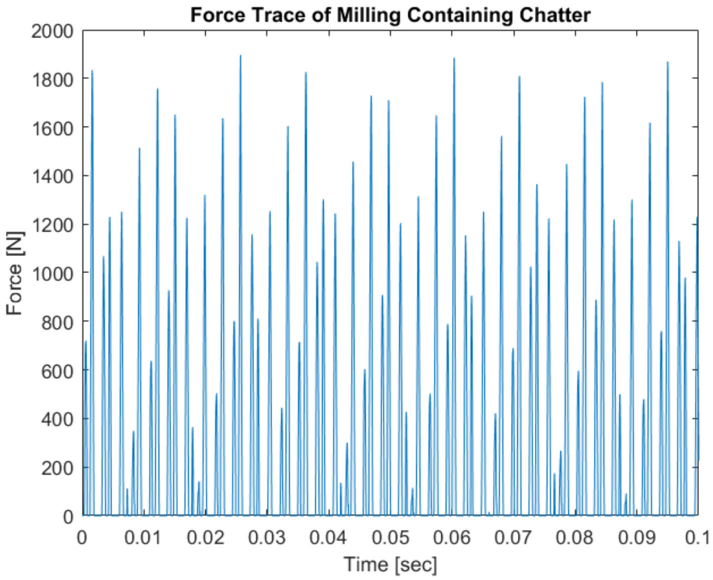


Figure 8: Simulated force trace of Fadal CNC Machine to show an example of chatter. (7500 RPM, 8 mm axial depth)

To account for the tool deflection, the chip thickness equation (Equation 3) is rewritten as:

$$h(\phi(t)) = f_t \sin(\phi(t)) + s(\phi(t-\tau)) - s(\phi(t)) \qquad (10)$$

where s is the surface variation due to tool deflection [mm], and τ is the time between tooth passings [sec.], i.e. s(t) is the instantaneous tool deflection and s(t-τ) is the regenerative deflection.

Milling process conditions under which chatter occurs can be represented by stability lobe plots. These plots are generated with a horizontal axis of spindle speed, and a vertical axis of axial depth of cut. The lobes are a function of the Frequency Response Function (FRF), which is the equation(s) that describes the vibration behavior of the tool end in response to a force applied at the tool end. Another common term for the FRF is the Transfer Function of the system. For the CNC milling system, the FRF is for the compliance dynamics of the system. This is modeled as a simple second order mechanical system, which was described in section 2.2.

An example of a stability plot is shown in Figure 9. In the plot, the areas above the lobes are regions of instability, while everything below is stable. Each lobe is a function of the number of complete sinusoidal cycles imprinted on the surface of the workpiece, labeled as N. For example, any cuts made around the farthest right lobe that leave an imprint on the workpiece, will only leave a fraction of a complete sinusoidal cycle on the cutting surface. This is noted as N = 0, which can be interpreted as there are zero complete cycles imprinted. Moving towards the left, each new lobe adds one more sinusoidal cycle. This explains why there are pockets of large axial depth stability at lobe intersections, since these are the areas where there is no phase shift between the present tool motion and the surface waviness left by the previous tool motion. The synchronous nature equates to a constant chip thickness and cutting force.

Figure 9 also shows the b_{lim} line, which is the minimum axial depth across all the stability lobes. This value is usually what is considered when choosing an axial depth to operate at, since any value below this line is independent of spindle speed.

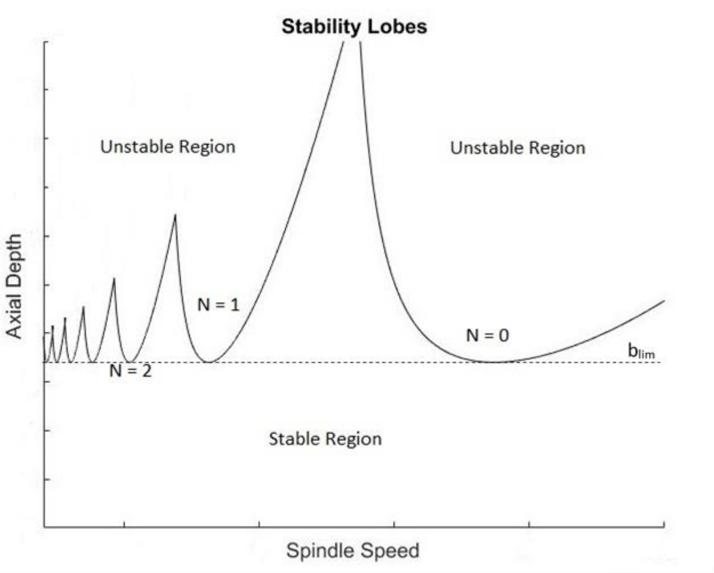


Figure 9: Typical stability lobe plot. Solid lines are lobes. Dashed line is axial depth limit (b_{lim}).

The two methods for generating stability lobe plots are by looking at a large number of experimental cuts, or through an approximate theoretical analysis using the compliance FRF of the CNC machine.

Theoretical stability lobes are often calculated by applying Tlusty's average tooth angle approach [8]. As the name suggests, this approach uses the mean tooth angle in the cut, which is the midpoint between the start and exit angles of cut. This is used because the force direction is time dependent, and assuming an average angle makes the system time invariant. Per Schmitz and Smith [6], generation of the lobes via this approach is a 6-step process.

First step is to determine the oriented FRF and valid chatter frequency range. The oriented FRF is calculated by summing the products of the orientation factors with their corresponding FRF (Equation 11). Milling occurs in the x-y plane, so the FRF of each the x-direction and y-direction are used. The orientation factors (μ_x and μ_y) are the trigonometric values needed to project the FRF's onto the average cutting force direction. They are calculated from the cutting force angle (which can be calculated from the cutting coefficients if not given) and the average tooth angle. The valid frequency range ($f_{chatter}$) is the set of frequencies where the real part of the FRF is negative. This limits the calculations to only where chatter is likely to occur.

$$FRF_{orient} = \mu_x FRF_x + \mu_y F\mathrm{R}F_y \tag{11}$$

Step 2 is to solve for ϵ, the phase between the current and previous tooth vibrations (eqn. 12), as determined from the Nyquist plot for the system [8].

$$\epsilon = 2\pi - 2\tan^{-1}\left(\frac{Re[FRF_{orient}]}{Im[FRF_{orient}]}\right) \tag{12}$$

Step 3 is to find the average number of teeth in the cut (N_t^*) for the selected radial immersion (eqn. 13). This calculation requires the total number of teeth on the tool (N_t), along with the start angle (ϕ_s) and exit angle (ϕ_e).

$$N_t^* = \frac{\phi_e - \phi_s}{360/N_t} \tag{13}$$

Step 4, calculate b_{lim}, which is the maximum axial depth attainable while avoiding chatter, over the valid frequency range (eqn. 14). This equation was obtained by solving for b_{lim} in the open-loop transfer function equated to -1 [8].

$$b_{lim} = \frac{-1}{2K_s Re[FRF_{orient}]N_t^*} \tag{14}$$

where;

$$K_s = \frac{K_{tc}}{\sin(K_{tc}/K_{rc})} \tag{15}$$

For step 5, select an N value and calculate associated spindle speeds Ω (eqn. 16). This equation is derived from the ratio of the chatter frequency to the forcing frequency, and the phase shift present. By solving for Ω, an equation is obtained that allows for the solving of Ω based on previously determined values. This calculation needs to be repeated for each N associated lobe desired.

$$\Omega = \frac{f_{chatter}}{N_t\left(N + \frac{\epsilon}{2\pi}\right)} \tag{16}$$

The last step is to plot spindle speed vs. axial depth for each N value. This process results in lobes like what is shown in Figure 9.

2.5: Summary

This chapter provides a thorough description of the two models used to simulate the milling machine operation. First is the linearized cutting force model, which is two coefficient-based equations that can calculate the radial and tangential forces experienced by the cutter. Second is the tool compliance model, which is represented as a standard two-dimensional, second order spring-mass-damper system. Appropriate modifications to the equations are described so they can be applied to a tool with multiple helical teeth. A comparison of experimental and simulation forces show the accuracy of the model when good model coefficients are used.

Machining chatter is also discussed in this chapter. The regenerative chip thickness equations that result in a positive feedback in the simulation are described. Also presented are stability lobes, a diagram of stable and unstable cuts plotted on a 2-D plane of spindle speed vs. axial depth of cut. A theoretical method to calculate the stability lobes using an average angle approach, originally developed by Tlusty, is also described.

Chapter 3: Milling Simulation

3.1: Introduction

This chapter describes the milling simulation used in the research. It starts with an overview of the three portions of the simulation: initialization, force model calculations, and system dynamics calculations. A complete description of the dynamic equations is presented, along with the methods used to obtain the system parameters.

Accurate simulations are useful in understanding and explaining the behavior of real world systems. The effect of parameter variation on the system can also be explored without the cost of experiments. However, an accurate simulation requires experiments to determine system parameters and to validate simulation results.

The cutting simulation used in this thesis contains a two-step process; (1) calculate the forces acting on the tool at an instant of time, and (2) use the forces to calculate the x and y deflections of the tool. The basis of the simulation is from Schmitz and Smith [6], which has been used in previous research at UNH [1].

The block diagram of the simulation, which is a more detailed version of the general diagram in Section 2.1, is shown in Figure 10. This diagram indicates that the linear force model, as described in Section 2.2, is used to calculate the forces experienced during cutting. The system compliance is assumed to be a second order system, the equations of which are described in Section 2.3.

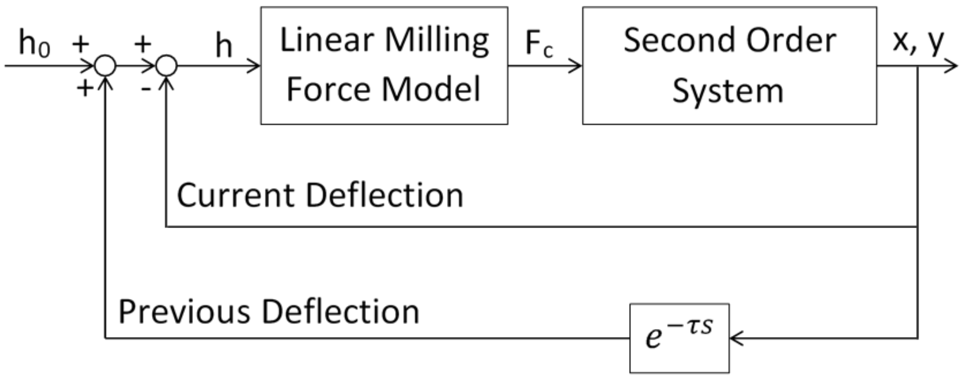


Figure 10: Block diagram for CNC system simulation.

3.2: Simulation Description

The simulation contains three main sections; input and initialization, force calculation, and displacement calculation. As shown in Figure 11, the first step initializes all the necessary constants and variables needed for the simulation. The constants are based on the inputs, and include all tool parameters (natural frequency, spring coefficient, damping ratio, tooth count, and diameter), cutting coefficients, and machine settings (feed rate, immersion, and axial depth). Dependent variables displacement, velocity, acceleration, and force are updated throughout the simulation, all starting with a zero-initial condition.

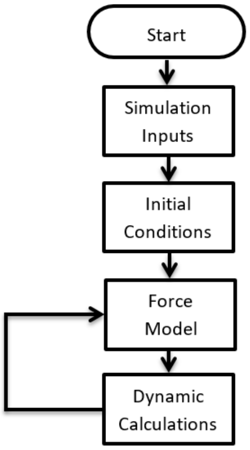


Figure 11: Overall flow chart for simulation.

The second section calculates the force experienced by the tool using the flow diagram shown in Figure 12. This section cycles through two loops for each time step; an inner loop for the axial slices of the tool, and an outer loop for the teeth on the tool. Forces are calculated using the process described in Section 2.2, with the addition of summing the forces calculated for each tooth.

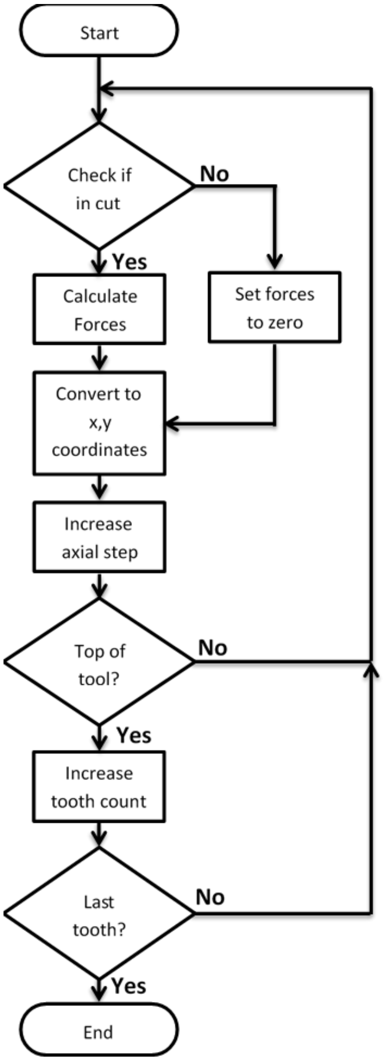


Figure 12: Flow diagram for force calculation.

The third portion calculates the displacement experienced by the end of the tool. As shown in Figure 13, this is done by calculating the acceleration, velocity, and position of the tool in the x and y-directions for each mode during each time step in the simulation.

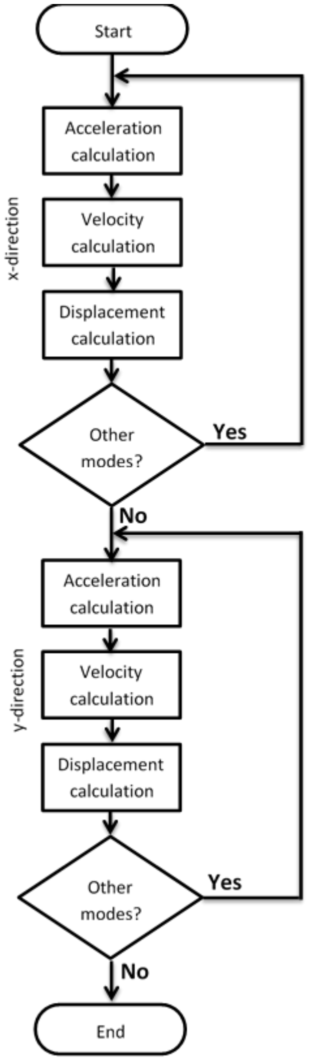


Figure 13: Flow diagram for system dynamics calculations.

The simulation works in equal time steps, with the variables considered as functions of the step index (i = 1, 2, …N), where N is the number of steps (final time) the simulation is to run. Solving Equations 8 and 9 for acceleration for index i results in:

$$\ddot{x}(i) = \frac{F_x(i) - c_x\dot{x}(i-1) + k_x x(i-1)}{m} \tag{17}$$

$$\ddot{y}(i) = \frac{F_y(i) - c_y\dot{y}(i-1) + k_y y(i-1)}{m} \tag{18}$$

These equations are valid assuming the previous time step (*i*-1) has already been determined. Since all initial conditions are assumed to be zero, calculations are not accurate for a number of time-steps.

To complete the state calculations, the velocities and displacements are determined using rectangular integration,

$$\dot{x}(i) = \dot{x}(i-1) + \ddot{x}(i)dt \tag{19}$$

$$x(i) = x(i-1) + \dot{x}(i)dt \tag{20}$$

$$\dot{y}(i) = \dot{y}(i-1) + \ddot{y}(i)dt \tag{21}$$

$$y(i) = y(i-1) + \dot{y}(i)dt \tag{22}$$

where *dt* is the time step [sec] between indices. The simulation loops through the force and dynamics calculations for as many time steps as requested by the simulation user.

The simulation program of Schmitz and Smith was written to handle higher-order dynamic models in the x and y direction (Figure 14) by using modal parameters. Modal parameters are another way to describe a system, and are obtained by decoupling higher-order mechanical systems into a series of independent second-order systems. The total response of the system is then calculated by summing the responses of the isolated systems. Each of these contributing responses is called a mode, and the use of modes helps simplify the calculations required in the simulation. Given the normal system parameters, the modal parameters can be calculated by finding the eigenvalues of the system. The calculation process is explained in more detail by Schmitz and Smith [6].

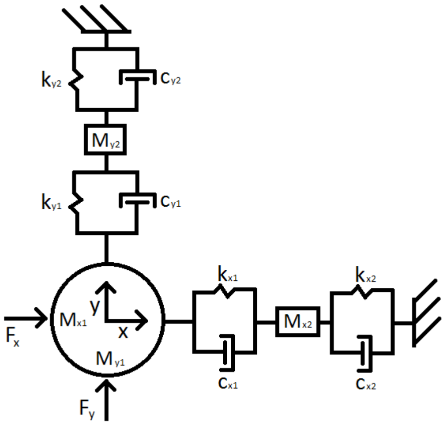


Figure 14: Diagram for double spring-mass-damper system, in the x and y direction.

3.3: System Parameters – Free Vibration Analysis

The dynamic parameters of the CNC machining model need to be determined to obtain accurate tool vibrations from the simulation. This includes natural frequency (ω_n), damping ratio (ζ), and spring constant (k) of the spring-mass-damper system in the x and y directions.

A common approach to finding the parameters of a single mode, second-order system is to analyze the free response to an impact. Methods have been developed and rigorously tested to reliably and accurately solve for the ω_n and ζ from the response. Unfortunately, the spring constant can’t be reliably determined from the response. Instead, k is calculated by analyzing force vs displacement of the tool, which should give an accurate value. Parameters must be determined in both the x and y-directions using the impact response and displacement-force response methods.

Impact testing was applied to the end of the cutting tool to obtain the required time trace. The equipment used was a PCB accelerometer (U353-B03), a PCB modal hammer (086C03) with a metal tip, two charge amplifiers, and a HP 35670A spectrum analyzer.

The modal hammer was equipped with a hard (steel) tip, with no added mass. The accelerometer was mounted to the end of the tool with wax as shown in Figure 15. The hammer and accelerometer were each wired through their own charge amplifier into the spectrum analyzer. The amplifier for the accelerometer was set to a gain of 1, while the hammer had a gain of 10.

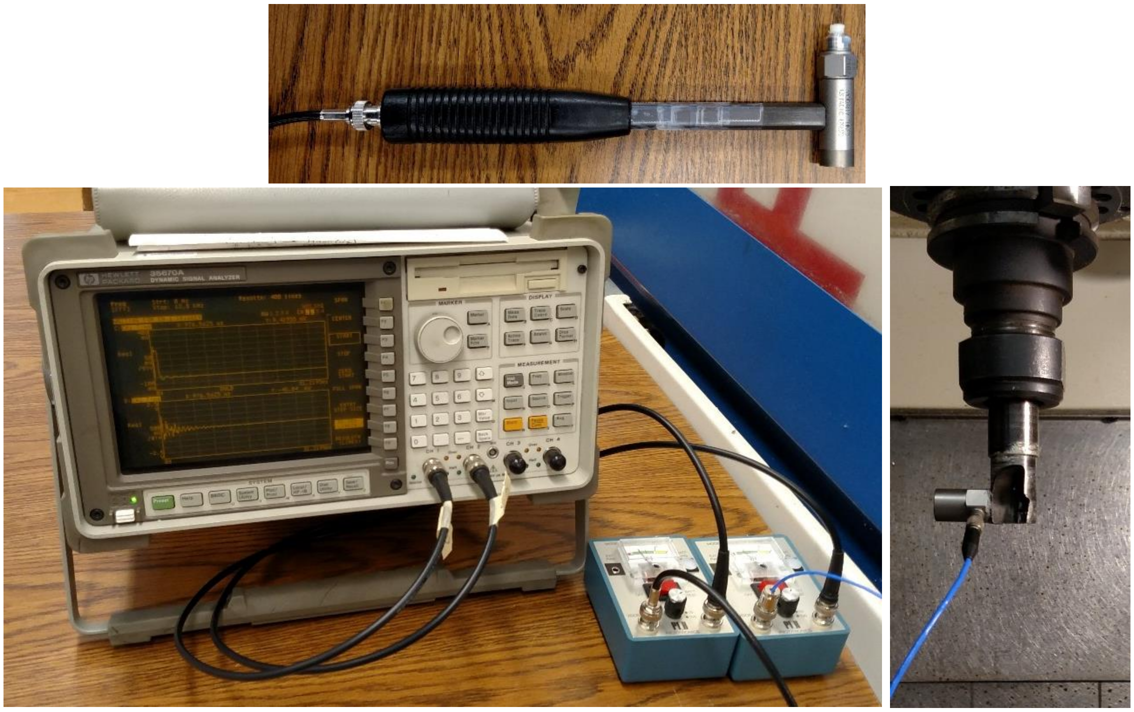


Figure 15: Tap test experimental setup.

The accelerometer was aligned in the x or y direction depending on the test, and the analyzer was set to collect the time trace of the accelerometer. A single sharp tap from the opposite side of the tool, as close to the end of the tool and as much in line with the accelerometer as possible. A sample of the accelerometer time trace collected is shown in Figure 16.

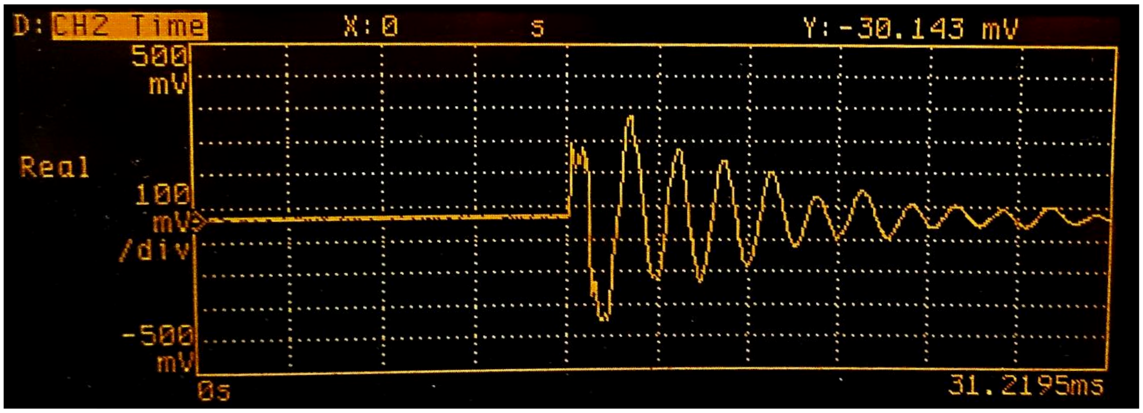


Figure 16: System response to impact.

The observed behavior in Figure 16 closely resembles the typical behavior for a second-order impact response. Based on this, the log decrement method, as described by Ogata [9], can be used to determine ζ. Equation 23 is a rearranged version of the one presented by Ogata, where n is the number of cycles in the calculation and T is the period (time to complete one cycle) of the signal. Multiple tap test calculations are made to get a range of ζ values, which are presented in Table 1.

$$\zeta = \frac{1}{\sqrt{1 + \left(\frac{2\pi}{\frac{1}{n} \ln\left(\frac{x(t)}{x(t + nT)} \right)} \right)^2}} \tag{23}$$

The damped natural frequency (ω$_d$) is simply obtained from the plot by taking the inverse of the time required to make a complete cycle. This is then converted to ω$_n$ by using Equation 24.

$$\omega_n = \frac{\omega_d}{\sqrt{1 - \zeta^2}} \tag{24}$$

The spring constant (k) is the ratio of force to displacement, so it can be calculated by applying a known displacement to the end of the tool, while the resulting force is measured. This was accomplished by placing a workpiece mounted to a Kistler 9257B dynamometer against the end of the tool. A measured displacement was applied using the machine's table, accurate to 0.0005" (0.01 mm), while the force was measured by the Kistler.

Figure 17 shows the measured force plotted against the measured displacement. The value of k is determined by finding the slope of the best straight line fit of the data set. These values are listed in Table 1, both of which seem reasonable based on previous testing at UNH [10].

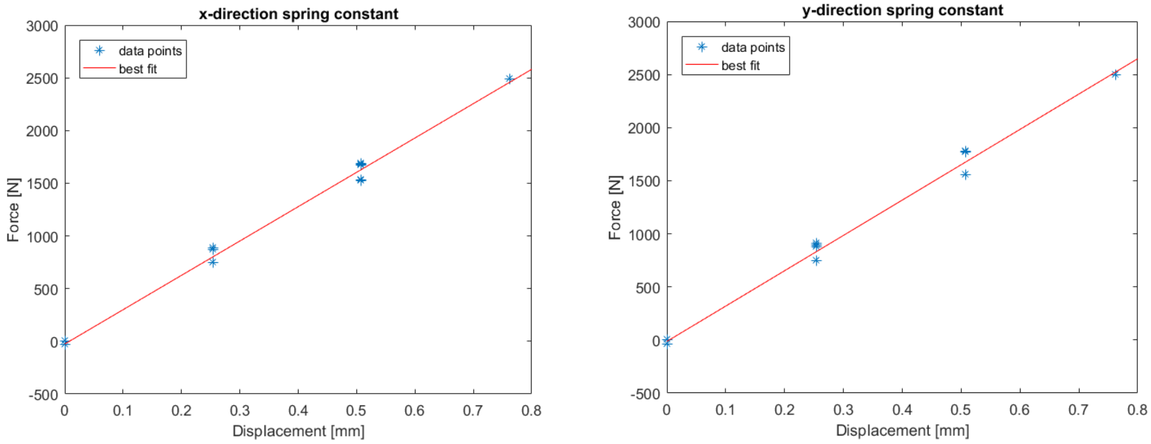


Figure 17: Kistler data fitted with a linear function. Slope of best fit line is the spring constant.

Table 1: Dynamic parameters found via application of free vibration and known displacement analysis.

	ω_n [Hz]	k [N/m]	ζ
x-direction	641	$3.258 \cdot 10^6$	.05 - .06
y-direction	641	$3.327 \cdot 10^6$	.05 - .06

3.4: System Parameters – Modal Fitting

Another method of system parameters identification is modal fitting, using the Frequency Response Function (FRF) of the system. As stated in Section 2.4, the FRF is the function that describes the vibration behavior of the tool end in response to a force applied to the tool end.

A common method to obtain the FRF is through impact testing. One version of this method is to use a modal hammer with a force sensor to impact the system, and an accelerometer to measure the vibration response of the system. The result is a FRF that relates input force to output acceleration.

The FRF can be visualized in many ways, with amplitude ratio and phase shift vs. frequency being the most common. For modal fitting, plots of the real and imaginary values vs frequency are needed. Four pieces of information are then used from the plots to calculate ω_n, ζ, and k. Shown in Figure 18, the four items are the peak value (A) and frequency value (ω_{n1}) on the imaginary plot, and the frequency value of the peaks of the real plot on either side of the imaginary peak (ω_2 and ω_3). Since this method is designed for modal parameters, the four items will need to be found for each mode desired.

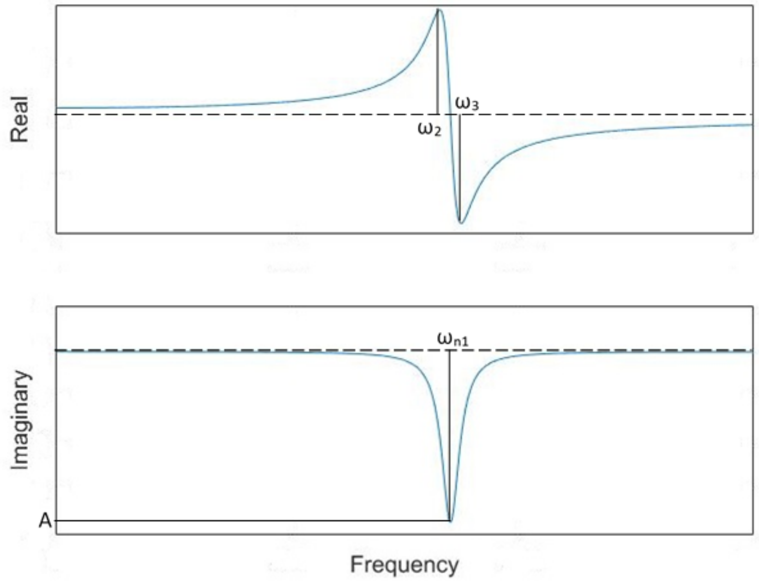


Figure 18: Visualization of modal fitting for one mode. Recreation of example plots shown in Schmitz and Smith [6].

Using modal fitting as explained in Schmitz and Smith [6], the FRF needs to relate displacement to force. Since these experiments use an accelerometer, the peak value that will be determined from the plot (A) will be in terms of acceleration in response to force (A_a). To convert it to its displacement equivalent (A_d), Equation 25 is used.

$$A_d = -\frac{A_a}{\omega_{n1}^2} \tag{25}$$

With these four values, the remaining parameters are then calculated. The first is ω_n, which is equal to ω_{n1}, available directly off the plot. The other two parameters are calculated using the equations below, derived and given by Schmitz and Smith [6].

$$\zeta = \frac{\omega_3 - \omega_2}{2\omega_{n1}} \tag{26}$$

$$k = \frac{-1}{2\zeta A_d} \tag{27}$$

As stated previously, the FRF for the tool is found using an impact testing method. The equipment and experimental setup for this is the same as for the free response analysis described in Section 3.3. However, the analyzer must be programed differently for the modal tests.

For modal testing, the spectrum analyzer was set for a 2-channel frequency response, displayed as real and imaginary plots. A force-expo window was applied to the signals, with values of 750 µs for force, and 4 ms for the decay. The force-expo window is a combination of 2 types of data windows, each applied to one signal. The force portion is a boxcar window over the initial impact, the length of which is the time the initial impact takes. This eliminates noise and ringing of the hammer, allowing for a better estimation of an impulse input. The response portion is an exponential decay window, the length of which is the time required for the signal to mostly settle. The main purpose for this is to smooth the data to the shape of a typical damped system. Additionally, the analyzer was set to be triggered by the hammer, and to average together ten data sets. This averaging smoothed the plots obtained, and improves the coherence of the data.

Once the analyzer is started, the taps are completed in the same way as described for the free response, including the separate testing of both the x and y-directions. The only difference is that the additional taps are required for the averaging. This results in a plot similar to that shown in Figure 19.

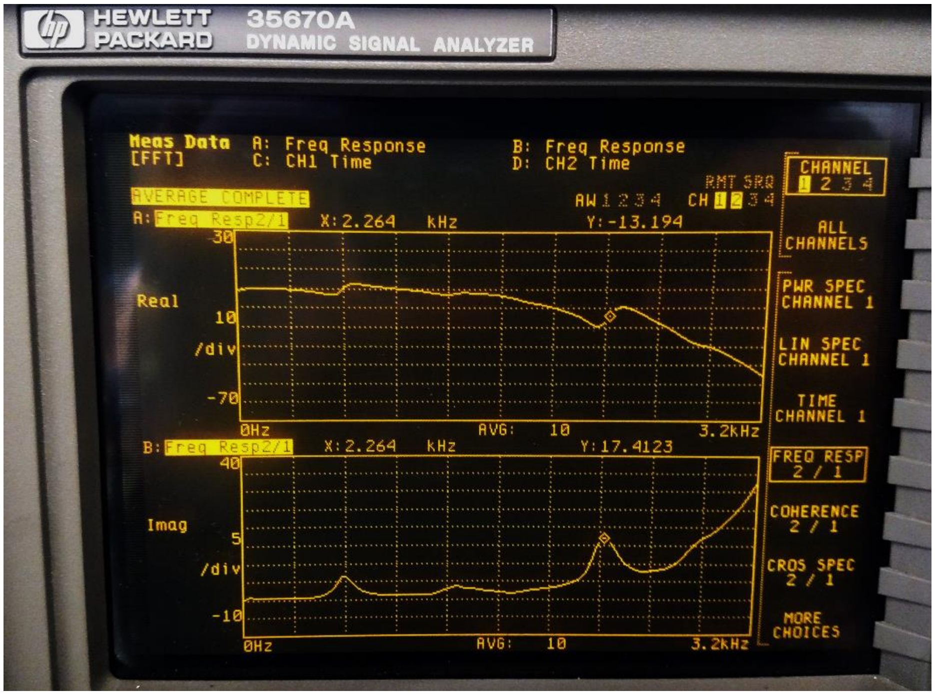


Figure 19: Sample output of HP Signal Analyzer.

To calculate the parameters of interest, the four pieces of information described earlier are needed. The values for the first two modes (first two peaks) are pulled from the plots generated in the experiment. The values listed in Table 2 are the average value calculated from two experimental runs. These values are used in Equations 25-27 to calculate the modal parameters, which are presented in Table 3.

Table 2: Modal fitting values determined from real-imaginary plots.

		A [V]	ω_{n1} [Hz]	ω_2 [Hz]	ω_3 [Hz]
x-direction	1st Mode	7.39	628	569	693
	2nd Mode	3.09	1104	1056	1152
y-direction	1st Mode	6.38	620	560	696
	2nd Mode	3.68	1328	1260	1368

Table 3: Dynamic parameters found via tap test. First two modes for each x and y directions presented.

		ω_n [Hz]	k [N/m]	ζ
x-direction	1st Mode	628	$1.21 \cdot 10^5$	.099
	2nd Mode	1104	$2.02 \cdot 10^6$	.043
y-direction	1st Mode	620	$1.22 \cdot 10^5$	.11
	2nd Mode	1328	$2.62 \cdot 10^6$	.041

Table 4 presents the parameters found through the free vibration analysis alongside the first mode parameters found through modal fitting. Both sets show similar values for ω_n, and have ζ values that are relatively close. The larger difference is in the k values where the force displacement method is an order of magnitude larger than the modal method. A decision must be made as to which parameter set to use.

Both methods have several sources of error. The major sources for Modal Fitting is the accuracy of plot measurement, and the compounding of possible errors in the calculations. The free vibration plot clearly shows that both the x and y vibrations are composed of multiple modes, not just the assumed single mode. Since we are interested in the motion of the tool end, the free vibration test yields approximate parameters that describe tool end vibration. It was decided to use the values found with free vibration analysis, mostly because the measurement interdependency in the calculations is smaller, which translates to less compounding of errors.

Table 4: Comparison of first mode parameters for free vibration and modal fitting methods.

	x-direction		y-direction	
	Free Vibration	Modal Fitting	Free Vibration	Modal Fitting
ω_n [Hz]	641	628	641	620
k [N/m]	$3.258 \cdot 10^6$	$1.21 \cdot 10^5$	$3.327 \cdot 10^6$	$1.22 \cdot 10^5$
ζ	.05 - .06	.099	.05 - .06	.11

3.5: Simulation Improvements

There are two major alterations made to the original code to help facilitate the cutting simulation and the determination of unstable cutting. First, the simulation is made time-step dependent rather than rotational-step dependent, and second, a check is added that terminates the simulation if the tool cutting forces become large enough to break the tool.

Stability of the milling system is analyzed through simulation using a large range of spindle speeds. Using a “steps per revolution” method is fine for most cases, however this method breaks down at low spindle speed since there is a point where the “sampling frequency” will be very low. From sampling theory, this low frequency increases the chance of aliasing and it increases error in the simulation since a simple rectangular integration scheme is employed.

To increase the simulation accuracy, a desired time-step is set. Using Equation 28, a value is calculated for the number of steps per revolution required to obtain the desired time-step. This value is then truncated to obtain an integer number of steps per revolution, and converted back to the actual simulation time-step using Equation 29.

$$Revolution\ step = \frac{60}{\Omega * (Time\ step)} \tag{28}$$

$$Time\ step = \frac{60}{\Omega * (Revolution\ step)} \tag{29}$$

Due to the instability of chatter, and the lack of displacement constraints in the program, it is possible for the cutting force to infinitely increase. A check was added for tool breakage by calculating the shear strength of the shank of the tool, assuming typical tool steel. This worked out to be approximately 1.9 MN for a ¾” (19.1 mm) diameter tool, and would be a worst-case scenario since the insert will most likely break before the tool shank. Once the tool shank fails, the simulation is ended.

3.6: Simulation Verification

The milling process simulation program is verified by first comparing simulation results to those from Schmitz and Smith [6], and second by comparing simulation results to actual cuts performed on the Fadal CNC machine. Both stable and unstable cuts are simulated.

A good initial confirmation is to compare simulation results from the revised simulation to those from Schmitz and Smith's [6]. The simulations are for four tooth slot cutting in aluminum, with a feed per tooth of 0.15 mm/tooth. The cutting coefficients are K_{tc} = 281 N/mm^2, K_{rc} = 281 N/mm^2, and K_{te} = K_{re} = 0. The mechanical system parameters used are equal in both x and y-directions with values ω_n = 500 Hz, k = 8e6 N/m, and ζ = .02. Both simulations are run with identical process parameters and constants.

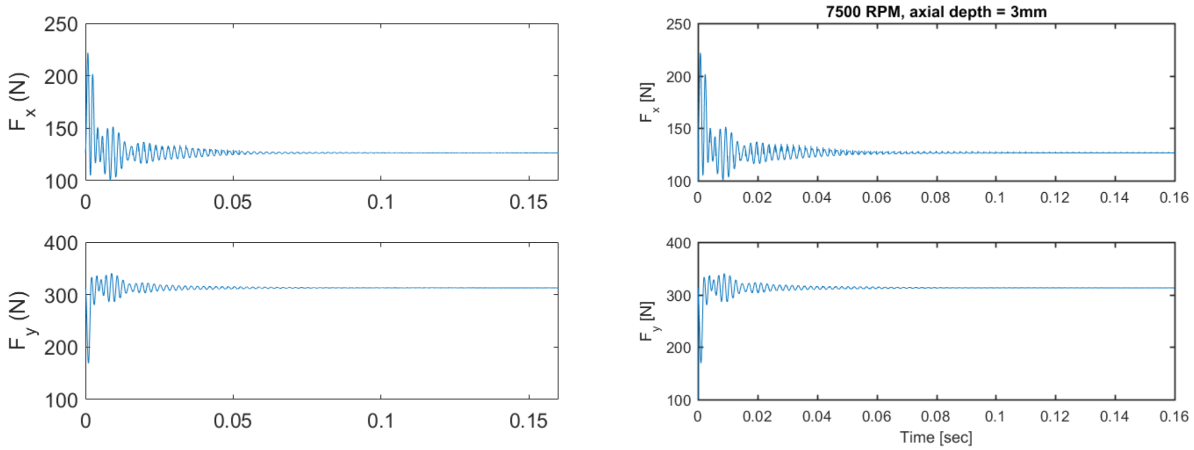


Figure 20: Comparison of stable force time traces; 7500 RPM, b = 3 mm. Left is simulation results from Schmitz and Smith [6, p. 142], and right is research simulation results.

The first set, shown in Figure 20, is of a stable cut; Ω = 7500 RPM, b = 3 mm. It is observed that the plots from both Schmitz and Smith [6] (left) and the research simulation (right) are nearly identical. They also clearly show the initial transient behavior caused by the tool's initial contact with the workpiece, followed by the decay to a steady force, as expected from a stable system. This shows initial confirmation to the accuracy of the research simulation, and demonstrates the ability to model steady cutting behaviors.

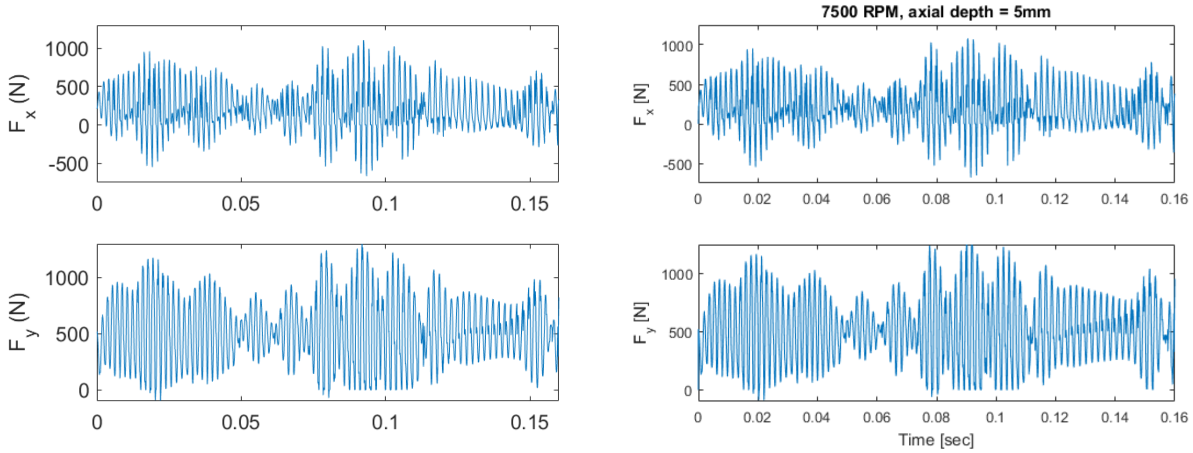


Figure 21: Comparison of unstable force time traces; 7500 RPM, b = 5 mm. Left is simulation results from Schmitz and Smith [6, p. 142], and right is this research simulation results.

The second set, shown in Figure 21, is of an unstable cut; Ω = 7500 RPM, b = 5 mm. Like the stable plots in Figure 20, the plots match well. The forces are very erratic, since the instabilities present in the system will not allow the forces to decay to a steady value. These plots give further confirmation of the research simulation's accuracy, and demonstrates the ability to model chatter.

The second confirmation compares simulation forces to experimental cutting data collected from the Fadal. The cut shown below (Figure 22) is quarter immersion in aluminum, at 2000 rpm, with an axial depth of 4 mm. The average chip thickness is set to 0.002" (0.05 mm), which equates to a feed rate of 25.32 in/min (0.64 m/min). The experimental data is smoothed using a 6th-order Butterworth low pass filter with a cutoff frequency of 500 Hz. The cutting coefficients used for the simulation data are K_{tc}

= 600 N/mm^2, K_{rc} = 120 N/mm^2, K_{te} = 20 N/mm, and K_{re} = 19 N/mm, along with the system parameters obtained by the method described in Section 3.5, which are $\omega_{nx} = \omega_{ny}$ = 641 Hz, k_x = 3.258e6 N/m, k_y = 3.327e6 N/m, and $\zeta_x = \zeta_y$ = .131. Tool runout, which is offset of the tool center, is included in the simulation for a better match. The simulation uses tooth-to-tooth runout, which means that the radial offset of each tooth is used as input, altering the chip thickness for each tooth. Through trial and error, runout for the three-tooth cutter was discovered to be approximately 5, -25, and 0 µm.

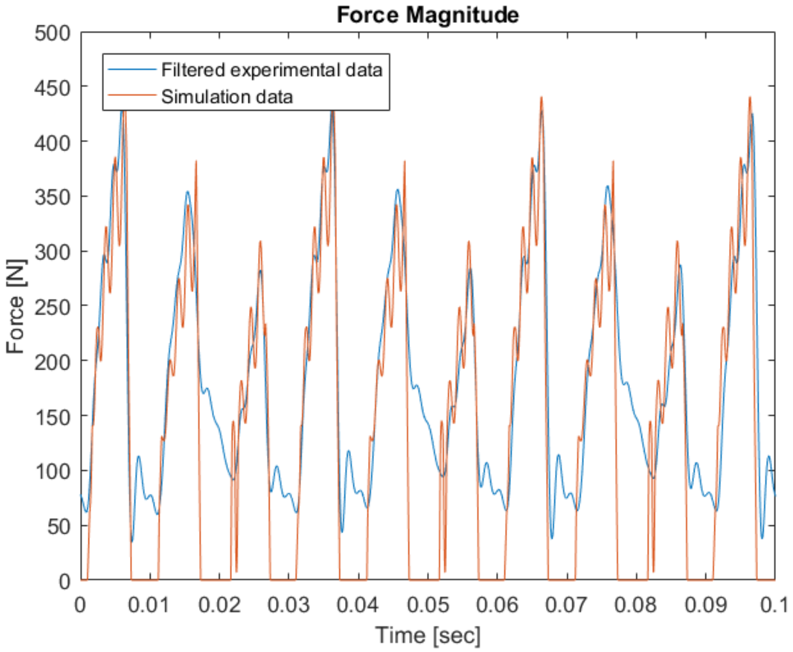


Figure 22: Stable force trace comparison between simulation and filtered experimental data; 2000 rpm, b = 4 mm.

By including runout, the simulation and experimental data presented in Figure 22 are a good match. The filter does remove most of the Kistler vibrations and noise, but evidence of its resonance still exists.

Experimental and simulation data is also compared for unstable cutting. The cuts are made at 5000 rpm with an axial depth of 7 mm. The experimental data was obtained and processed in the same way as described for the stable cuts, but due to the higher spindle speed, has a feed rate of 63.3 in/min (1.61 m/min). The simulation uses the same parameters as the stable cuts.

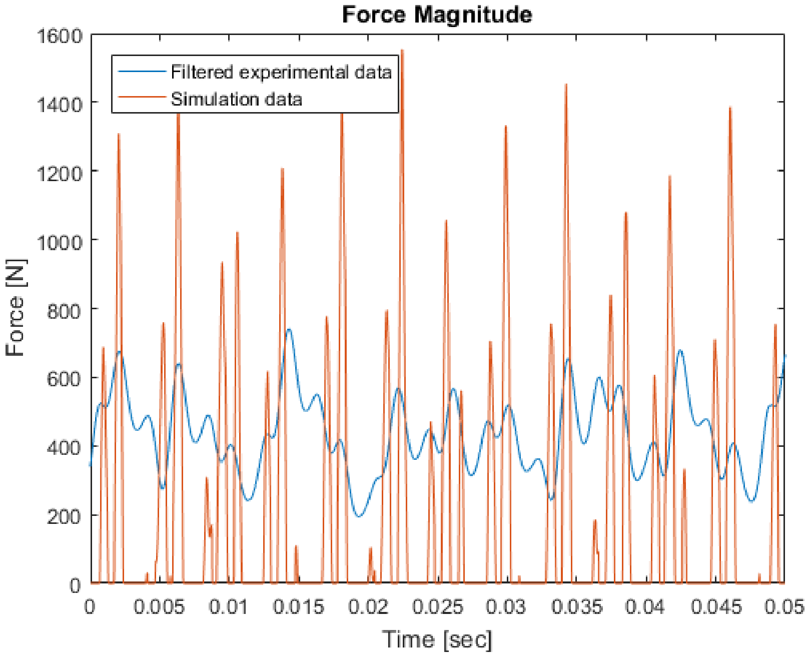


Figure 23: Unstable force trace comparison between simulation and filtered experimental data; 5000 rpm, b = 7 mm.

Figure 23 shows that there are similarities between the force traces of the experiment and simulation. The data sets don't appear to match very well, but show strong evidence of chatter by the non-repeating force peaks and the tool moving in and out of the material, indicating very large vibrations. Also, due to the complexity of the milling process, and the chaotic nature of chatter, the data sets are not expected to match perfectly. Chatter is evident in both traces because the force does not repeat itself form one tool cycle to the next. To attain a better match would require the use of a much more sophisticated model.

3.7: Summary

This chapter describes the MATLAB-based simulation used in this research. The simulation steps through time, calculating forces and displacements experienced by the tool for each time step. Total forces are calculated by the equations described in Section 2.2, looping to ensure all cutting teeth and axial slices are accounted for. Tool motion is calculated by applying Eulerian integration on the equations described in Section 2.3.

The two methods to experimentally determine the spring constant (k), natural frequency (ω_n), and damping ratio (ζ) of the Fadal milling machine are detailed. These methods are free vibration response and Modal Fitting. The first method consists of two separate tests. The first test collects data from an accelerometer mounted at the end of tool, in response to an impact, and the second test measures forces experienced when a known deflection is applied to the end of the tool. The Modal Fitting method calculates the parameters using values measured from the real and imaginary plots of a Frequency Response Function, obtained through impact-response testing. The parameters found through both methods are presented and compared. They all fall within a reasonable range, however the parameters found through free vibration response are considered more reliable and are used in the simulation of this thesis.

The two major improvements made to the simulation are presented. The first converts the simulation from revolution-step dependence to time-step dependence. The second adds a check that terminates the program if cutting forces exceed 1.9 Mega-Newtons, breaking a ¾” (19.1 mm) tool.

The revised simulation program is validated for both stable and unstable cases, through comparison with results from both Schmitz and Smith [6] and filtered force data from the Fadal. Schmitz and Smith’s results are nearly identical to those from the simulation, providing good initial verification.

Stable cuts from the Fadal provide a close match with the simulation, especially when tool runout is included in the simulation. The unstable cut data does not match, but since both show evidence of chaotic behavior caused by tool fluctuation, it is concluded that reasonable simulation of chatter can be obtained. This all shows that the simulation can accurately portray the real-world system’s behavior.

Chapter 4: Chatter Detection

4.1: Introduction

This chapter discusses past work in milling chatter detection, with a focus on the method that uses once per revolution sampling. The algorithm developed for this research is described, and presented with the simulation results used to evaluate the algorithm.

Algorithms to accurately determine the existence of chatter during a machining operation could help make machining safer and more cost effective. Knowledge of chatter could be used to make the operator aware of the potentially damaging condition, or it could be used in a closed-loop control system that could automatically adjust the machine, returning to a safe and stable state. Accomplishing this requires a robust algorithm that can autonomously analyze force, displacement, or acceleration data, and return a determination of cutting stability.

4.2: Previous Work

Many methods have been explored for online chatter detection. These cover a range of sensors, and the data processing techniques with each method having advantages and disadvantages.

Since chatter results from the vibration caused by the tool-workpiece interaction, many of the most popular methods attempted have relied on analysis of this interaction, mostly by way of the noise generated by the tooth engagement. This method involves using a microphone to collect noise data [2]. Chatter is detected when there is a positive jump in noise levels. The major advantage to this method is that it is fairly inexpensive to implement, and requires low computation power. The disadvantage comes from applying it in a manufacturing environment, since the ambient noise from other activity in the area will most likely be picked up by the microphone.

There have been attempts at measuring tool deflection directly. This is not straight forward, since the desired measurements are at the end of the tool, which is constantly in motion and in contact with the workpiece during milling operations. There are a few sensors that could be used in this situation, but they all require a smooth surface and/or a clear path of view to the tool. This can be reasonably done in a laboratory setting, but almost impossible for an industrial application. One such setup was used by Altintas [3], where a proximity sensor was aimed at a steel sleeve fitted just above the cutting section of the tool. This method would provide a direct measurement of tool vibration resulting from chatter, however sensor line of sight would make this method impossible in most industrial milling application. Perhaps new sensors will make this method feasible someday.

The Fourier Transform of the cutting force has been explored as another way to determine the existence of chatter [11]. This method involves using a Fast Fourier Transform (FFT) routine to generate a frequency plot. Since chatter is defined as an unwanted vibration, it will show noticeable spikes on the plot, making this a valid way to show the presence of chatter. The large disadvantage to this method is the data processing time that would be required. Most current FFT routines require a fair amount of computation to complete the transform, which when added to the algorithms required to locate and process the FFT spikes, can lead to delays and errors in determining the initiation of chatter, and any resulting actions.

A more complicated chatter detection algorithm was proposed by Ma and Melkote [4]. They apply several filters to a force trace collected with a Kistler 9257B dynamometer, and analyze it for the presence of chatter. They performed several cuts on a CNC Mill, and the algorithm successfully detected chatter when it was present. Ma and Melkote concluded that this method works for most of the cases tested, and uses a fraction of the computing power in comparison to the FFT detection method. The downside is that the multiple steps increase the complexity of the algorithm.

An alternate approach to on-line chatter avoidance is through process planning using accurate simulation models of the CNC process. Cutting conditions that give rise to chatter in the simulation can be changed during the process planning phase. Several researchers, such as Altintas [12] [13], have pursued this work by developing increasingly complex models.

The approach taken in this research is based on work by Schmitz [2], where sound data is sampled once per revolution and evaluated for chatter. This algorithm is discussed in more detail in the next section.

4.3: Original once per rev deflection algorithm

The once per revolution sampling method described in Schmitz [2] is based on Poincaré mapping techniques for determining system stability. A Poincaré map is a cross section of a system's orbit (sampling of points from a specific position on the cycle), which means its only applicable for cyclical behaviors. Since z-direction motion is negligible, the motion plot will just be x vs y tool motion points measured at the same tool location angle for every tool rotation. Figure 24 shows the x-y plots for all tool rotation angles for 25 cycles of both a stable and unstable simulated cut. Displacements are given for continuous motion of the tool as it rotates and the starred points on the plots show the once per revolution sample, which can be considered the Poincaré map for the system. It can be easily seen that for stable cutting, the sampled points converge to a single point, while the unstable plot has a larger, more elliptical shaped map. Figure 25 shows the stable cut of Figure 24 in more detail.

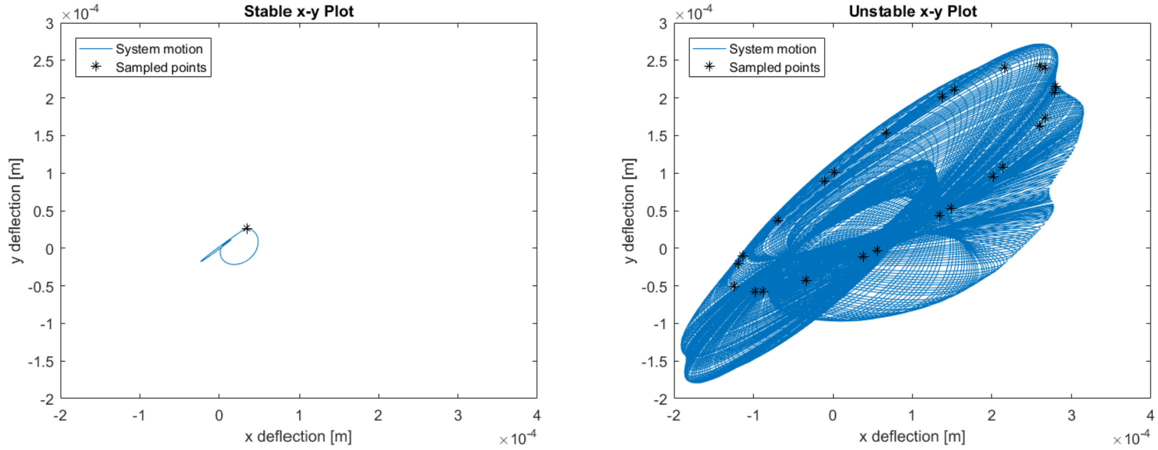


Figure 24: x vs y deflection plots of 25 revolutions. Both plots are quarter immersion, 5000 RPM. Left is a stable cut (b = 2) and right is an unstable cut (b = 6).

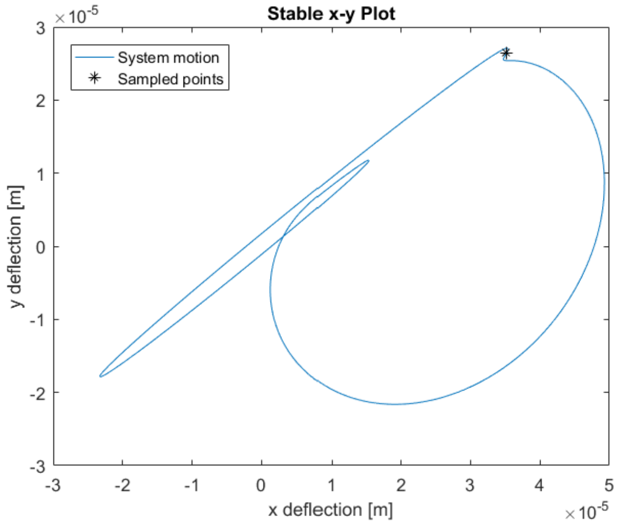


Figure 25: A zoomed in version of the stable x-y plot in Figure 24.

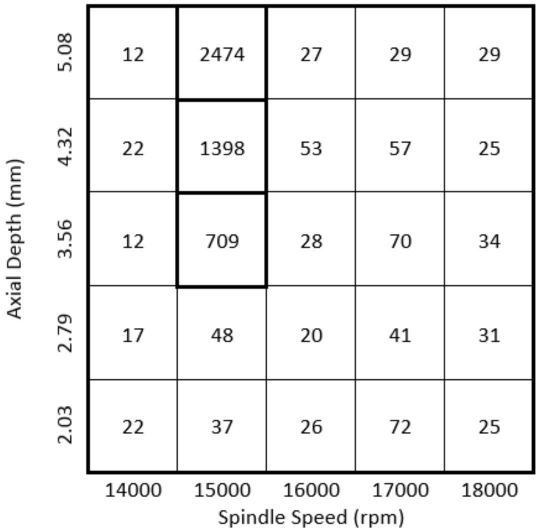
Measuring displacements while milling is very difficult, so Schmitz used a unidirectional microphone as data input to the Poincaré technique. Once per revolution sampling was triggered by a visual sensor aimed at a piece of reflective tape on the tool. This ensured the audio data is collected is at the same point on the tool for each revolution, since the Poincaré technique requires that all the data points be of the same portion of the orbit.

The possibility of chatter is indicated on the Poincaré plot by the spread of once per rev points. To quantify this, Schmitz calculated the variance (σ^2) by:

$$\sigma^2 = \frac{\sum_{i=1}^{N}(x_i - x_m)^2}{N-1} \tag{30}$$

where x_m is the average of the data set, and N is the size of the data set.

To better show the spread of variance values calculated from experimental microphone data, Schmitz organized them into a grid relating the variances to their respective spindle speed and axial depth, which is shown in Figure 26. The values in the bolded sections are a spike in variance, which coincided with observed chatter. From this, he concluded that variance is related to stability, and could possibly be used for detection.



Axial Depth (mm)	14000	15000	16000	17000	18000
5.08	12	2474	27	29	29
4.32	22	1398	53	57	25
3.56	12	709	28	70	34
2.79	17	48	20	41	31
2.03	22	37	26	72	25
	Spindle Speed (rpm)				

Figure 26: Variance values [mV²] from cutting test completed by Schmitz [2].

4.4: Evaluation of original once per revolution deflection algorithm

Schmitz's original once per revolution deflection algorithm was coded in MATLAB, and tested using the milling simulation described in Chapter 3, which uses cutting coefficients of K_{tc} = 600 N/mm^2, K_{rc} = 120 N/mm^2, K_{te} = 20 N/mm, and K_{re} = 19 N/mm, and system parameters of $\omega_{nx} = \omega_{ny}$ = 641 Hz, k_x = 3.258e6 N/m, k_y = 3.327e6 N/m, and $\zeta_x = \zeta_y$ = .131. This allows all variables to be available for analysis, signals to be cleaner than experimental data, and the ability to more readily explore algorithm sensitivities to cutting conditions. Results were compared to those from Schmitz to ensure accuracy of the coding.

The MATLAB based algorithm was written to collect 25 once per rev samples of displacement magnitudes after the initial transients settled out. The variance of the sample set is then calculated, and used to make a decision about the cut's stability. The determination of chatter is based on variance values chosen by the user, which means tuning of the algorithm is required if cutting conditions (i.e. radial immersion, workpiece material, cutting tool, etc.) are altered.

Tuning of the algorithm requires several data sets from a mix of stable and unstable cuts. The easiest method is to choose a spindle speed, and increase axial depth starting from well below the axial depth chatter limit, b_{lim}. The effectiveness of this method increases with the number cuts made at various axial depths and spindle speeds. This is due to the increasing ease of patterns being observed, which results in better tuning of the algorithm. After some basic tuning, the algorithm proves to be a viable way to detect chatter, although there appear to be a number of false positives for chatter.

False positives can be disruptive to the cutting process and are highly undesirable. Based on this, improvements are needed in the detection algorithm to give a more accurate determination of chatter presence. This is further complicated by the difficulty in determining chatter from cutting forces and cutter deflections since currently, the most definitive definition of chatter is the wavy surface left on the workpiece.

4.5: Deflection algorithm improvements

While developing the algorithm, the data contained a level of noise that did not appear to be related to chatter. Based on this, it was decided that straight variance would not be the best representation for the presence of chatter. Instead, the algorithm will use the magnitude of the difference between the x and y deflections of a sampled point, and its respective sample from the previous revolution. This is calculated by:

$$diff(i) = \sqrt{(x(i) - x(i+1))^2 + (y(i) - y(i+1))^2} \tag{31}$$

where *i* is the sample index. Just like previously, the variance of the differences is calculated, and related to chatter through a tuning process.

Taking the differences yields information about the change in displacement between cycles rather than the overall displacement of each cycle. This ensures that constant tool deflections do not affect the calculation, and lessens the effect of longer time-scale changes. The magnitude ensures the total motion of the tool is considered, not just one direction. This also means that the calculation is independent of the coordinate system, meaning the axis can be defined in alternative orientations, and the algorithm will still hold.

The standard once per revolution deflection algorithm is normally visualized through x vs y plots. Sample plots for both stable and unstable milling is shown below (Figure 27).

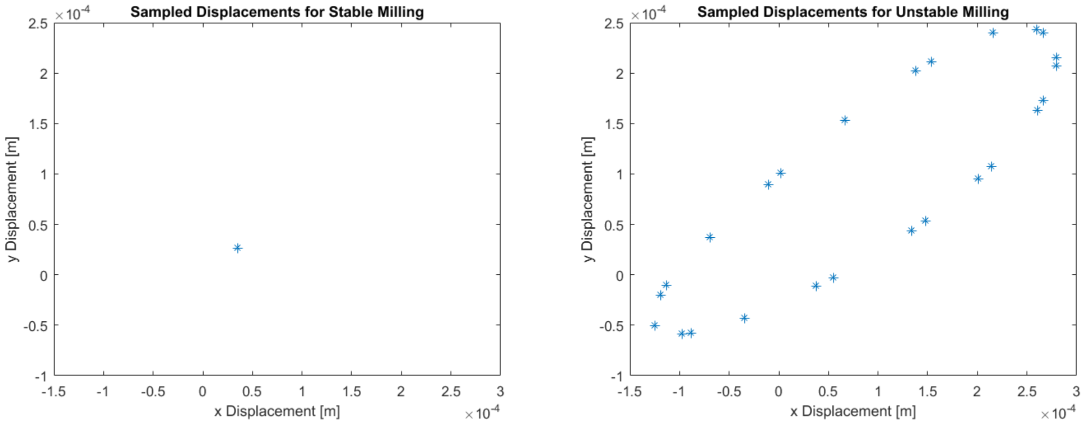


Figure 27:Plots are of once per rev sampled, x vs y deflections from simulated cuts. Both are aluminum, 5000 rpm, with a 3-tooth cutter. Stable cut (left) is axial depth of 2mm and unstable cut (right) is axial depth of 6mm.

For comparison, the results of the difference-magnitude algorithm for the same cutting conditions used in Figure 27 are plotted below (Figure 28 and Figure 29). Since these results are the magnitudes of difference, it is more difficult to visualize what they represent. The basis of the approach is that instabilities cause a larger, more erratic range of motion in comparison to stable operation. By plotting the calculation results vs. the sample number, a good visualization of the behavior can be obtained.

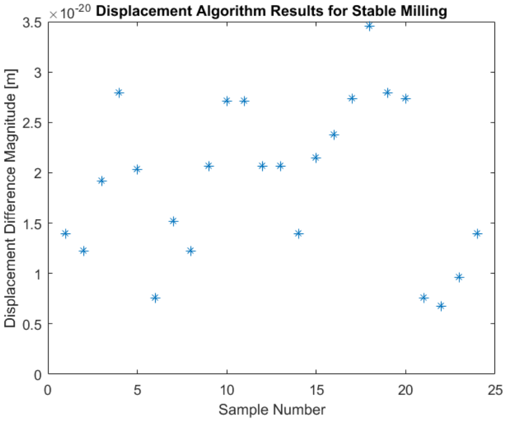


Figure 28: Results of algorithm for stable cut described in Figure 27.

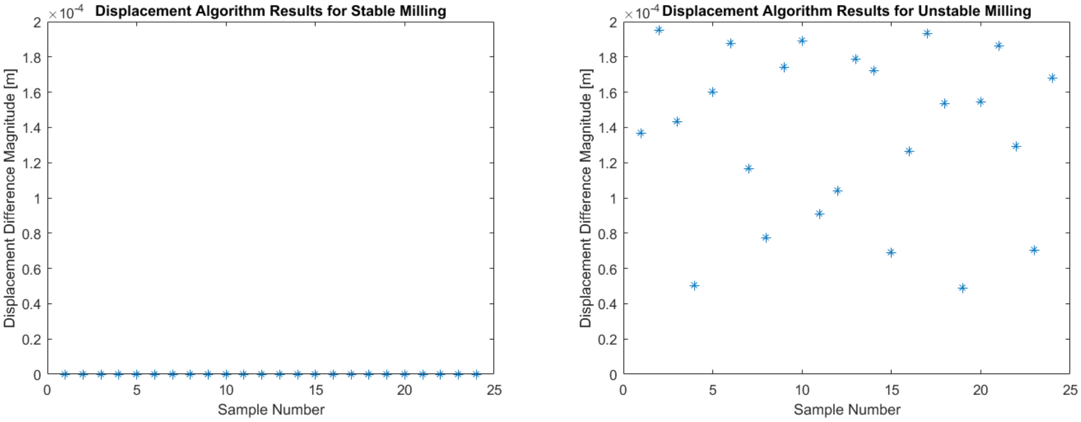


Figure 29: Results of the difference-magnitude algorithm applied to displacement data for unstable cut described in Figure 27 (right). Stable plot (left) is same as Figure 28, but scaled to match unstable plot.

Looking at Figure 29, it can be easily seen that the values calculated from the unstable cut are magnitudes larger than the stable cut. Comparing the zoomed in stable plot (Figure 28) to the unstable plot, the tighter consistency in consecutive data points for the stable cut contrasts with the more wildly varying points of the unstable cut. From this, it can be concluded that there is validity in using the difference-magnitude algorithm on displacement data for detecting the presence of chatter.

To show the improvement of the new algorithm over Schmitz's, analysis of displacement data for 4250 rpm at b = 4.75 mm was completed. Figure 30 shows visualizations for each of the algorithms. The left plot, which is the visualization for Schmitz's, shows the tool oscillating between two points at values spread enough to question the system's stability. This is in contrast to the difference magnitude values in the right plot, which form a very consistent, straight line. The line indicates that the cut is definitively stable.

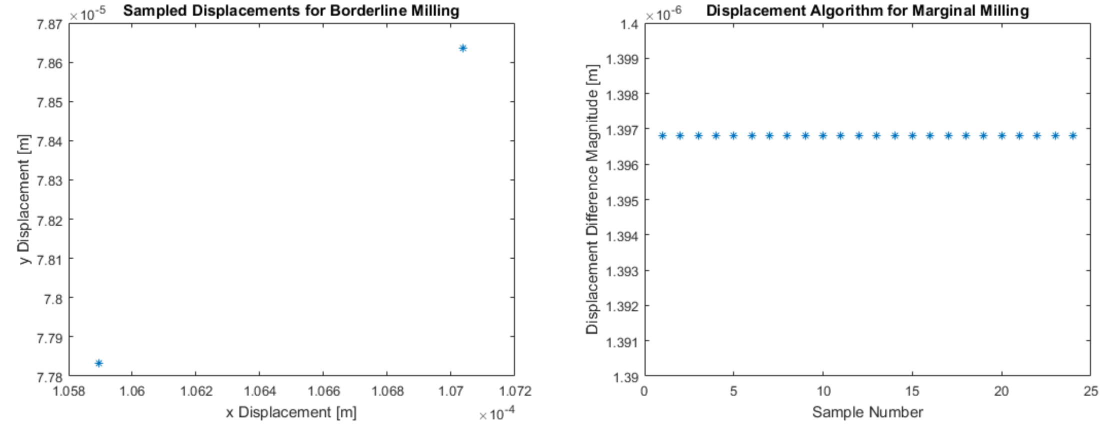


Figure 30: Displacement results for 4250 rpm, b = 4.75mm. The left plot contains the visualization of Schmitz's algorithm, which shows the tool oscillating between two points. The right plot shows the difference magnitude values, which form a very consistent, straight line.

4.6: Use of Force data

Tool deflections are a proven method of chatter detection, based on both past research and simulation results presented here. The major issue with this method, as discussed in Section 4.2, is implementation for online detection in manufacturing environments. This is also why Schmitz [2] explained his method through displacement, but experimentally implemented his algorithm using a microphone.

Tool deflection sensors require either a clear line of sight or constant contact with the tool. Most of these sensors require a smooth, clean surface to either be aimed at or attached to, neither of which are available during machining operations. Contact sensors could interfere with the machining process, and could be easily damaged by the machine motion. Indirect systems could be kept at a safe distance, but would be blocked by material chips and coolant. For these reasons, switching to the more easily and reliably obtained force data makes sense. A 3-axis dynamometer, used in this research, or a tool based dynamometer such as the Smart tool [14], are both sturdy enough to be used in an industrial setting. The same once per revolution sampling and calculations described in Equation 31 for displacement can be applied to the force data.

4.7: Confirmation of algorithm applied to simulation force data

The displacement based plots shown in Figure 29 can be recreated using force data as shown in Figure 31. The same trend of the stable cut samples having a much tighter grouping than the unstable cut is observed in the plots.

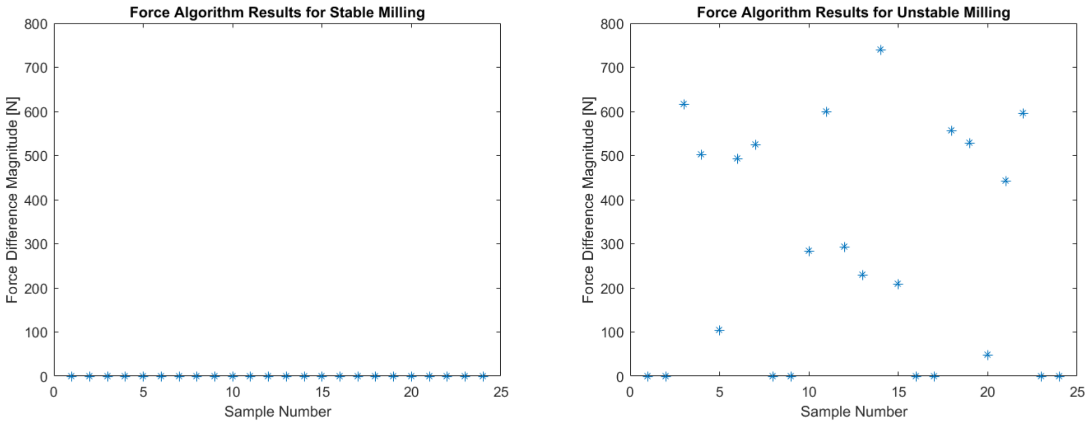


Figure 31: Results of force differencing algorithm for stable cut (left) and unstable cut (right).

Figure 32 is the stability plot developed by applying the chatter detection algorithm to simulation force data for a variety of axial depths and spindle speeds. Each algorithm determination is represented by a point. By looking at a sampling of force traces, reasonable variance limits are chosen. The ones used for these plots are less than 10 for stable, greater than 5000 for unstable, and between these values for marginal.

The simulation is of up milling with a quarter immersion of a ¾" (19.1 mm) tool, at a feed rate of .0016 in/tooth (0.04 mm/tooth). The cutting coefficients used are K_{tc} = 600 N/mm^2, K_{rc} = 120 N/mm^2, K_{te} = 20 N/mm, and K_{re} = 19 N/mm, with system parameters of $\omega_{nx} = \omega_{ny}$ = 641 Hz, k_x = 3.258e6 N/m, k_y = 3.327e6 N/m, and $\zeta_x = \zeta_y$ = .131. The theoretical lobes drawn on the plot were calculated using the average tooth angle approach described in Section 2.4 [6].

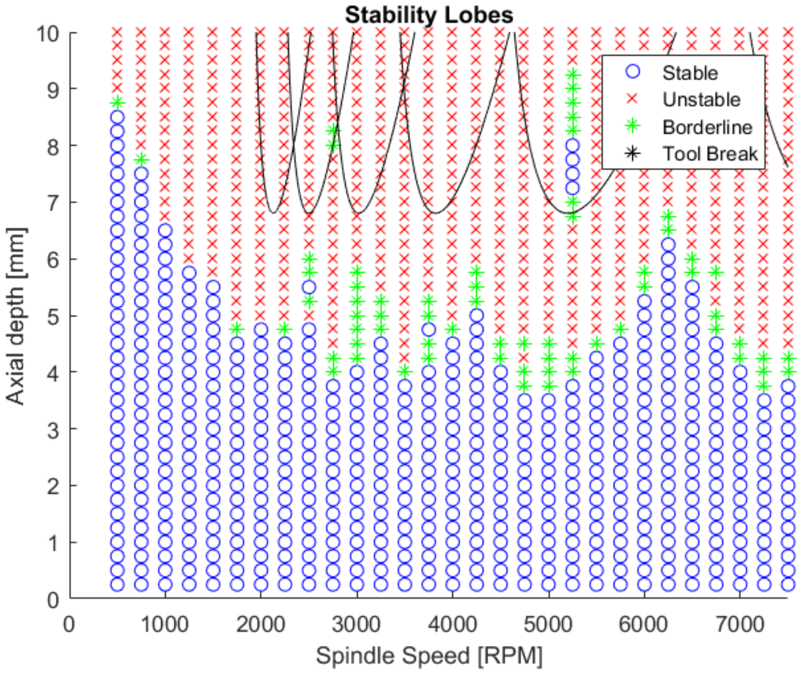


Figure 32: Simulation generated stability plot, determinations by difference-magnitude algorithm applied to force data. Average angle approach for theory.

The stability lobe plot of Figure 32 shows discrepancy between the theoretical lobes and the algorithm produced lobes. This discrepancy appears to be a simple axial depth and spindle speed shift in the lobes, staying fairly consistent from peak to peak, with the number of peaks nearly matching. The shift is approximately 3 mm in the axial depth direction, and 250 rpm in the spindle speed direction. Some shift would be expected since the theoretical lobes require a number of simplifying assumptions. The large shift in axial depth is somewhat unexpected and explored in Chapter 6.

Based on visual observations of the force traces, the chatter detection algorithm results appear to be reliable. Accuracy of the algorithm is explored in the next chapter using experimental cuts.

4.8: Summary

This chapter describes previous work in milling chatter detection, and this research’s contribution to that work. The past methods covered include the use of noise, displacement, and force data to determine the presence of chatter. The methods applied to these data sets include once per revolution sampling, Fast Fourier Transform (FFT) analysis, and multi-step filtering calculations. The development of more complex cutting models was also discussed, even though they are more suited for preprocess planning than online detection.

Schmitz’s once per revolution method [2] is presented as the basis for the algorithm developed in this research. This method is based on Poincaré mapping techniques, which states that a cyclical system’s stability is related to the amount of spreading present in a cross-section of the cycle. Schmitz used cuts performed at a variety of axial depths and spindle speeds to show that the variance of tool displacements sampled at the same point in the cycle is much greater in unstable cuts in comparison to stable cuts.

The improved detection algorithm is then described. The algorithm uses the same data sampling described for Schmitz’s method, but takes the differences between consecutive data points, and then determines the magnitude of the differences. Examples of clear cases for both stable and unstable simulated cuts are presented.

Initially, all algorithm results are based on the displacement of the tool. Given the impracticalities associated with monitoring tool deflections, the use of force data is explored. Just as with displacement, force sampled cases are shown for both stable and unstable cuts. This illustrates that forces experienced by the tool show similar behaviors to tool deflection, and therefore, can be easily implemented into the algorithm.

A stability lobe plot generated using simulation data analyzed by the algorithm is presented. This plot also contains stability lobes determined through theory. These two components don’t agree in axial depth limit, but show similar shapes.

Chapter 5: Experimental Work

5.1: Introduction

This chapter discusses the experimental testing used to validate the detection algorithm. The results from the experiments are presented, evaluated, and compared to simulations.

Simulation and theory are a cost-effective way to analyze the milling process. However actual experimental data is required to validate the simulation results and to explore the complicated nature of chatter. Stable and unstable (chatter) cuts are explored through experiments on the Fadal CNC machine. Results are used to confirm the chatter detection algorithm, and to develop a stability lobe diagram for comparison to one generated by simulation. The cuts are selected to ensure a variety of stability levels, i.e. stable, marginally stable, and unstable.

5.2: Experimental cuts

Experimental cuts are made on aluminum 6061 for comparison to simulation results (discussed in Section 3.6) and to evaluate the chatter detection algorithm. All cuts are up milling with a radial quarter tool immersion. Spindle speed and axial depth of cut is varied to explore stable, unstable, and marginally stable cuts.

The experimental cuts are made on a Fadal EMC 3-axis milling machine with a PC based open architecture controller from MDSI (Figure 33). The machine contains a horizontal, 2-axis table to which the workpiece is mounted, along with a vertical moving spindle. The tool holder and tool used in the experiment is shown in Figure 34. The ¾” (19.1 mm) diameter tool was equipped with three Sandvik Carbide inserts (R390-11 T3 08E-NL H13A).

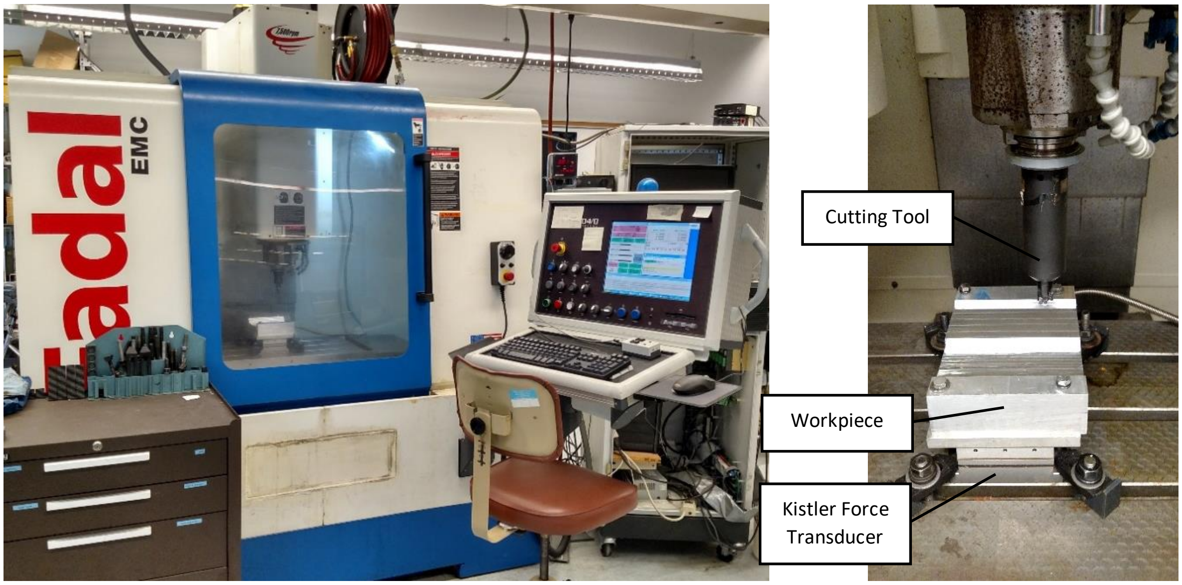


Figure 33: Fadal CNC milling machine. Overall setup (left) and close up of spindle and work piece mounted on Kistler force dynamometer (right).

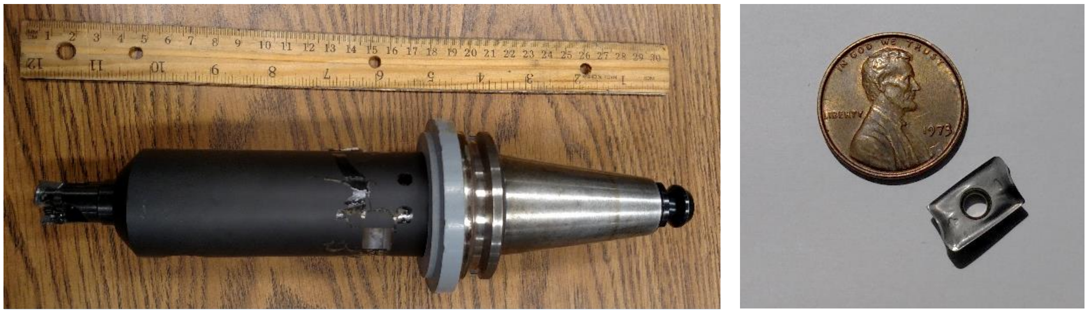


Figure 34: Cutting tool used for experimental cuts (left). Sandvik Carbide insert (right).

Figure 35 shows the hardware component diagram for the Fadal CNC Machine. This diagram shows the data flow from the CNC machine and three external sensors back to the control PC. All the data can be used in process monitoring and control, or in this case, collection for further analysis.

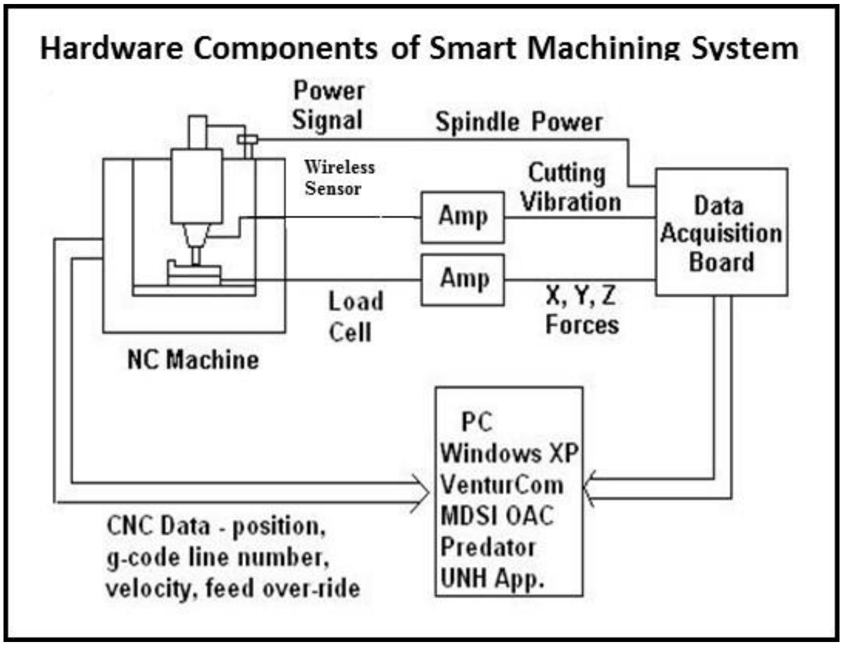


Figure 35: Hardware components of the CNC system: machine, controller, and sensors. [15]

The load cell, a Kistler 9257B Multi-Component Dynamometer, is the only sensor used in the research (Figure 36). The force signal is amplified by a 3-channel charge amplifier from Kistler Instrument Corporation, sampled by a high-speed data acquisition board, and then stored in the CNC control computer. Data collection is completed by a C# program developed in house, which samples and saves force data for x, y, and z-direction signals, along with time stamps.

Figure 36: Kistler 9257B Multi-Component Dynamometer [16].

The experimental cuts are made using a constant average chip thickness of 0.002” (0.05 mm), based on standard speed and feed tables for aluminum. The desired average chip thickness of 0.002” (0.05 mm), equates to a feed per tooth (FPT) of 0.0042 in/tooth (0.11 mm/min) for a quarter immersion cut. The spindle speed is changed throughout the experiments, so the feed-rate must be calculated for each spindle speed (Equation 32).

$$feedrate\ [in/min] = FPT[in/tooth] * Number\ of\ teeth * \Omega[rpm] \tag{32}$$

To ensure the reliability of applying the once per rev sampling, a sampling rate of 360 samples/rev must be maintained. This means the sampling rate must also be calculated with each spindle speed (Equation 33).

$$samping\ rate\ [Hz] = \ \Omega[rpm] * (360[samples/rev]/60[sec/min]\ ) \tag{33}$$

The goal of the experiments is to develop a full stability plot of spindle speed vs axial depth with a full grid of data points. To ensure safe operation of the machine, i.e. no chatter, the initial cuts are made at 2 mm for an even distribution of spindle speeds. The axial depth is then increased, and cuts for another even distribution of spindle speeds is made. This process is continued until chatter is found. Once chatter is found, no more cuts are made at that spindle speed for greater axial depths, since chatter is expected to continue, which could put the machine at greater risk of damage.

The determination of chatter in the experiments is determined through observations of the cut as they are made. Noise levels are monitored by ear for chatter, and the process is interrupted if it becomes excessive. After the cut ends, either through completion or interruption, the roughness of the workpiece surface is evaluated. Since chatter is defined by the wavy surface, the roughness of the surface is used as a final determination of chatter.

Table 5 lists the spindle speed and axial depth for all the experimental cuts, along with the feeds and sampling rates required for each spindle speed. This also includes the additional cuts made at spindle speeds between stable and unstable points, which help pinpoint the exact point of transition to instability.

Table 5: Table of cuts made for experiments. Contains feeds and speeds that maintain a chip thickness of 0.002" (0.05 mm), along with sampling speed required to maintain 360 samples/rev.

Spindle Speed [RPM]	Axial Depth [mm]	Feed [in/min]	Sampling (360 samples/rev) [kHz]
2000	4, 6, 7	25.32	12
2125	7	26.90	12.75
2250	7, 8	28.49	13.5
2375	8	30.07	14.25
2500	4, 7, 8	31.65	15
3000	2, 4, 6, 7, 8	37.98	18
3125	8	39.56	18.75
3250	7, 8	41.15	19.5
3375	7	42.73	20.25
3500	4, 7, 8	44.31	21
3625	7, 8	45.89	21.75
3750	7, 8	47.48	22.5
4000	2, 4, 6, 7, 8	50.64	24
4500	4, 7, 8	56.97	27
4625	8	58.55	27.75
4750	8	60.14	28.5
5000	2, 4, 6, 7	63.30	30
6000	6	75.96	36

To ensure the quality of the data collected, several considerations are made while running the experiments. First, a minimum of 5 minutes is given between cuts to allow time for the tool and workpiece to return to room temperature, decreasing the effect of heat since no coolant is used. Also, between each cut, chips are cleared from the workpiece area and the existing workpiece surface, potentially very rough, is smoothed by a light end mill cut.

After data is collected, the chatter detection algorithm, detailed in Section 4.5, can be applied. The data consists of x and y voltages, proportional to force, and a time stamp for each set of data. The data is imported into MATLAB along with the time vector. The force signals are then converted from voltage [V] to force [N], and used in the detection algorithm by sampling the forces based on the time vector. The algorithm determines if the cut is stable or unstable (chatter) based on tuning values given by the user.

Tuning values for chatter determination require a couple of stable and unstable test cuts. Variance values can be determined that separate stable and unstable cuts. If cutting conditions are not changed, except for depth of cut, the tuning values should be useful. Further precision can be made by looking at more data sets, especially ones near the transition zone.

5.3: Experimental Stability Results

This section contains results and analysis from the experimental cuts based on conditions in Table 5. These cuts are all done on the Fadal CNC milling machine, using up-milling at a quarter immersion, and an average chip thickness of 0.002" (0.05 mm). The cuts are used to generate stability lobe diagrams based on observation and on the chatter detection algorithm. The detection algorithm is evaluated by comparison of the stable, unstable and marginally stable points on the two plots.

To better visualize the stability results, several force traces and surface finish photos are shown below. All the force traces are filtered with a 6th-order Butterworth low pass filter with a cutoff frequency of 500 Hz to reduce the resonance effect of the Kistler load cell. The surface generated by the cut is bordered by yellow lines to highlight the cut profile of interest. These examples include stable, unstable, and borderline cases.

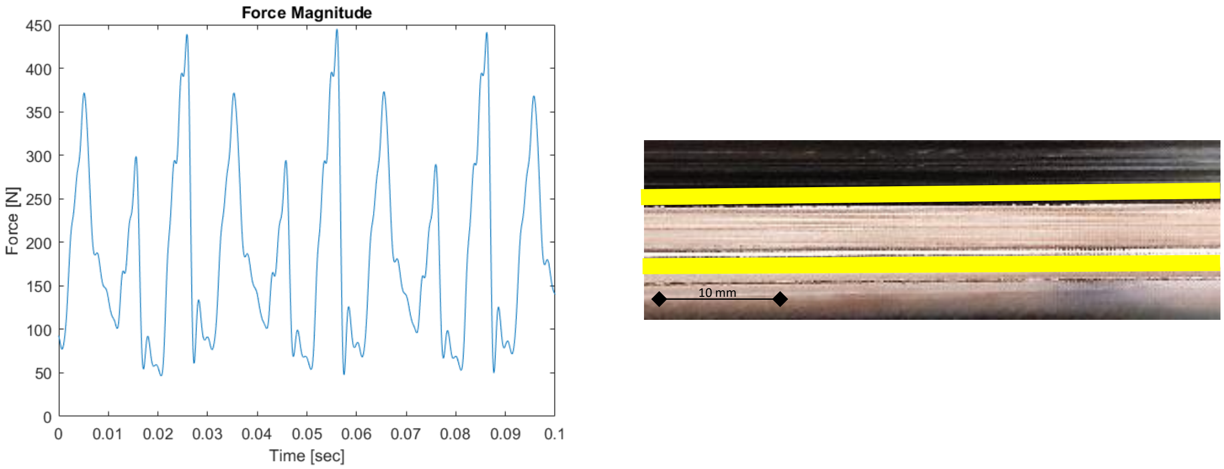


Figure 37: Stable cut force trace (left), and photo of cut surface (right). 2000 rpm, b = 4 mm

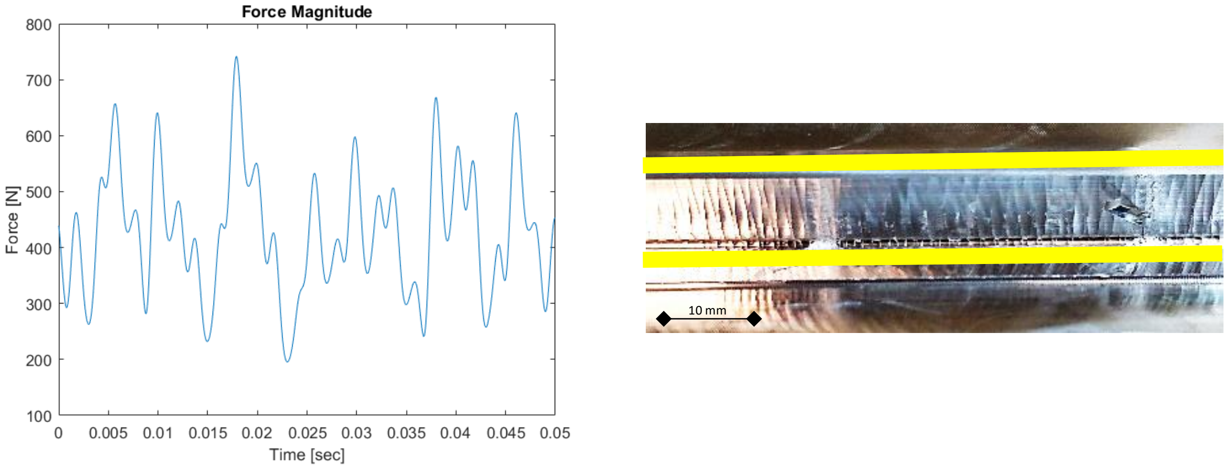


Figure 38: Unstable cut force trace (left), and photo of cut surface (right). 5000 rpm, b = 7 mm

Figures 37 and 38 are clear examples for stable and unstable cutting. The stable cut (Figure 37) has a clean and repetitive force trace, which coincides with the smoothness of the workpiece surface. This contrasts with the unstable cut (Figure 38) which has a force trace that is very erratic, i.e. non-repetitive, with much higher peak forces. The workpiece surface generated by the unstable cut is visibly rough, showing clear evidence of fully developed chatter.

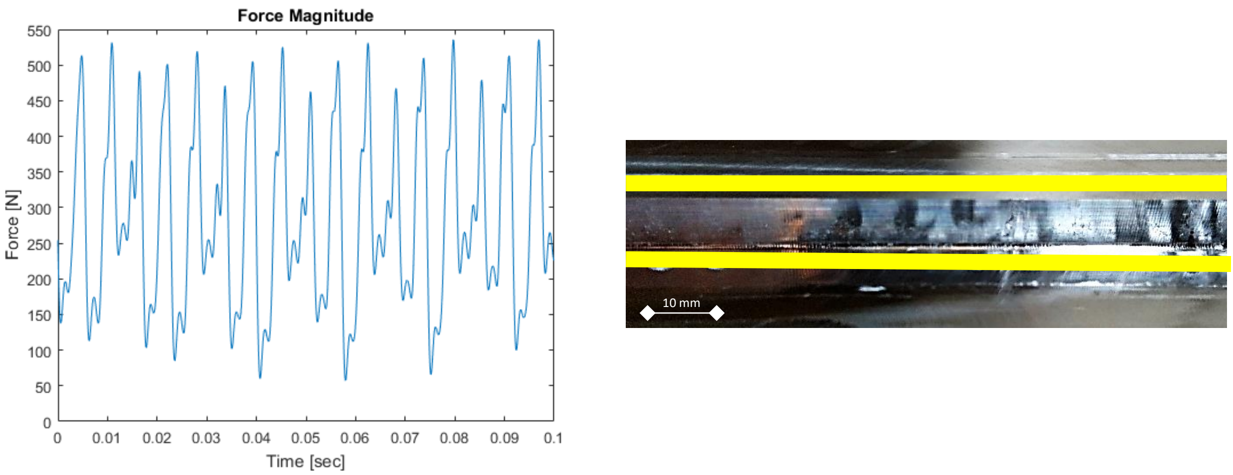


Figure 39: Cut determined to be marginally stable through observation, but stable by algorithm. Force trace (left), and photo of cut surface (right). 3500 rpm, b = 7 mm

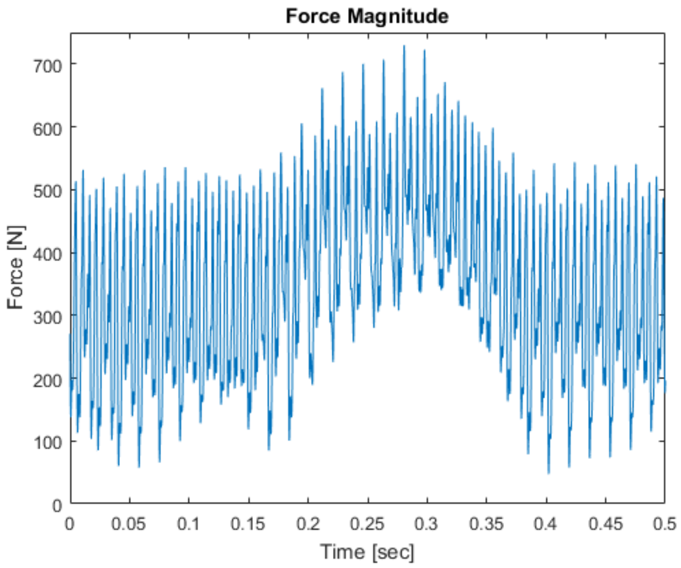


Figure 40: Zoomed out version of force trace in Figure 39.

Figure 39 shows a cut where the system is transitioning from stable to unstable, making the determination of the existence of chatter unclear. For this cut, the stability is categorized as borderline based on observation, since the surface shows a low frequency vibration that causes an alternating pattern of rough and smooth surfaces (which isn't shown in the force trace). On a larger time-scale (Figure 40), it is more readily seen that a low frequency vibration exists, with the force going in and out of chatter. The chatter detection algorithm determined this to be stable since it looks at the change between consecutive revolutions, which is slow enough to make categorizing it as chatter very difficult.

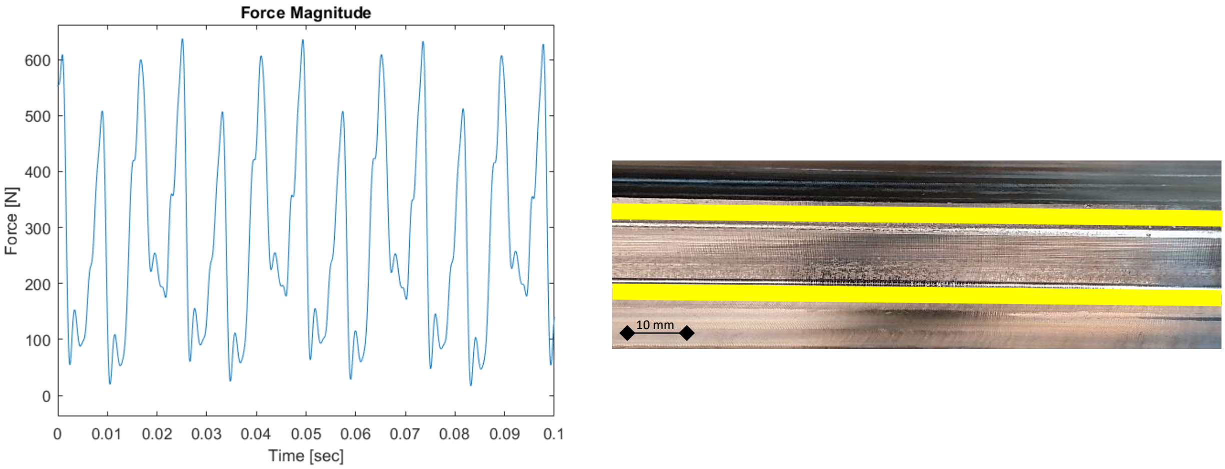


Figure 41: Cut determined to be stable through observation, but unstable by algorithm. Force trace (left), and photo of cut surface (right). 2500 rpm, b = 8 mm

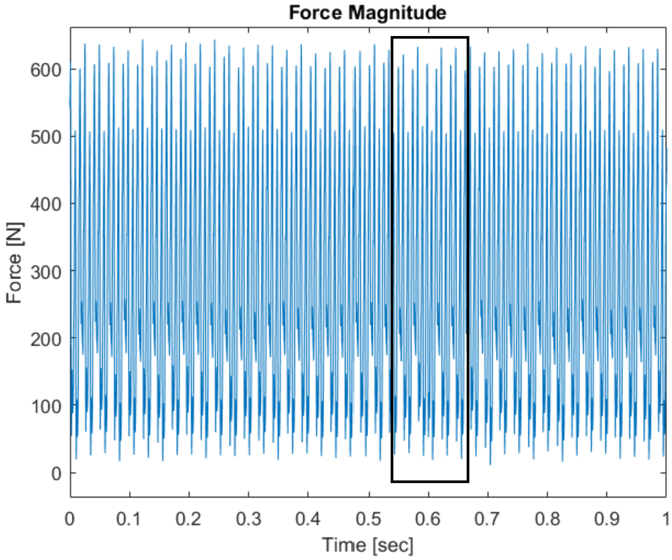


Figure 42: Zoomed out version of force trace in Figure 41. Boxed area is a section of force trace where repetitive nature is interrupted.

The cut results shown in Figure 41 is another example where the difference between stable and unstable can be unclear to the detection algorithm. Based on the smooth appearance of the surface and the repetitive nature of the force trace, the cut is categorized as stable by observation. When analyzed by the algorithm, the cut is determined to be unstable. This false detection is based on the forces shown in Figure 42. Most of the trace looks to be consistent, but there are areas where the pattern changes, one such area being outlined in black. This seems to be a false positive, but even if this is not fully developed chatter, it is interesting that the algorithm still picks up on small changes that could be the first hints of unstable conditions.

The first stability lobe plot (Figure 43) shows the stability results of the experiments as determined by observations of the noise and workpiece surface generated during cutting. The second stability plot (Figure 44) shows the stability results of the experiments as determined by the chatter detection algorithm.

The stability plots are graphs of cutting stability for a range of axial depths and spindle speeds. The stability points are plotted using a blue circle for stable cuts, a red x for chatter, and a green star for marginally stable cuts. Additionally, Figure 44 contains a black diamond to represent the one data set that was corrupted, and unusable for the algorithm.

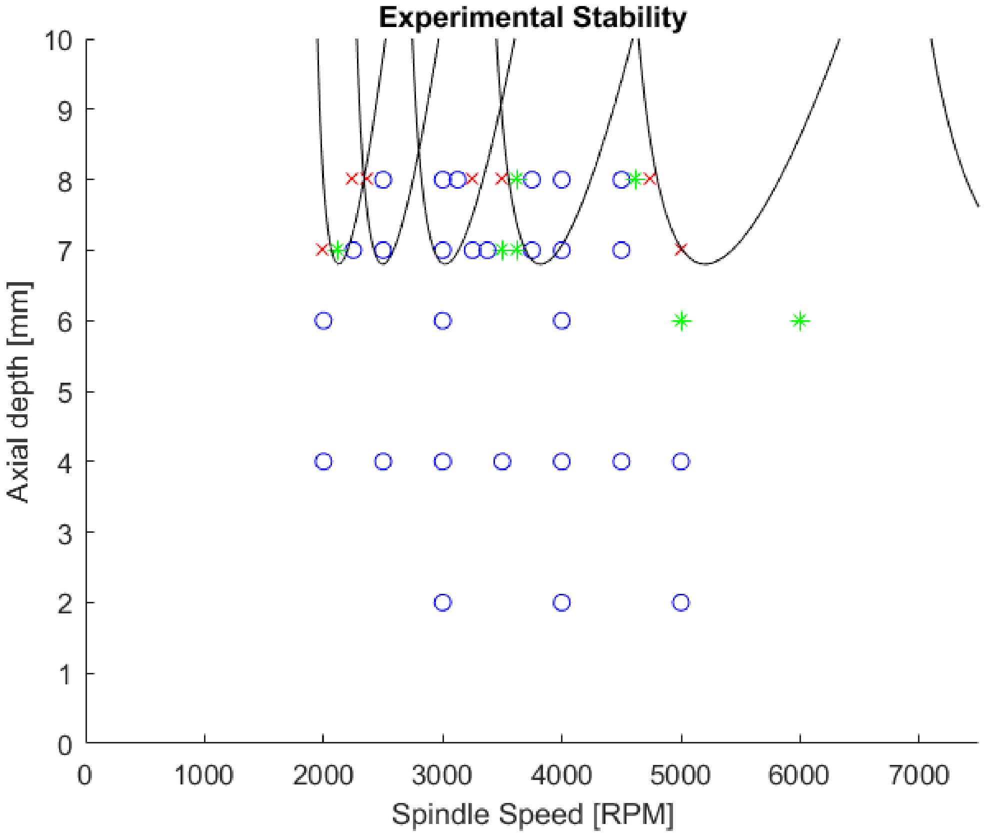


Figure 43: Experimental chatter results based on observed status of cut. Blue circle = stable, red x = chatter, and green star = borderline. Also included is the theoretically determined lobes, plotted as black lines.

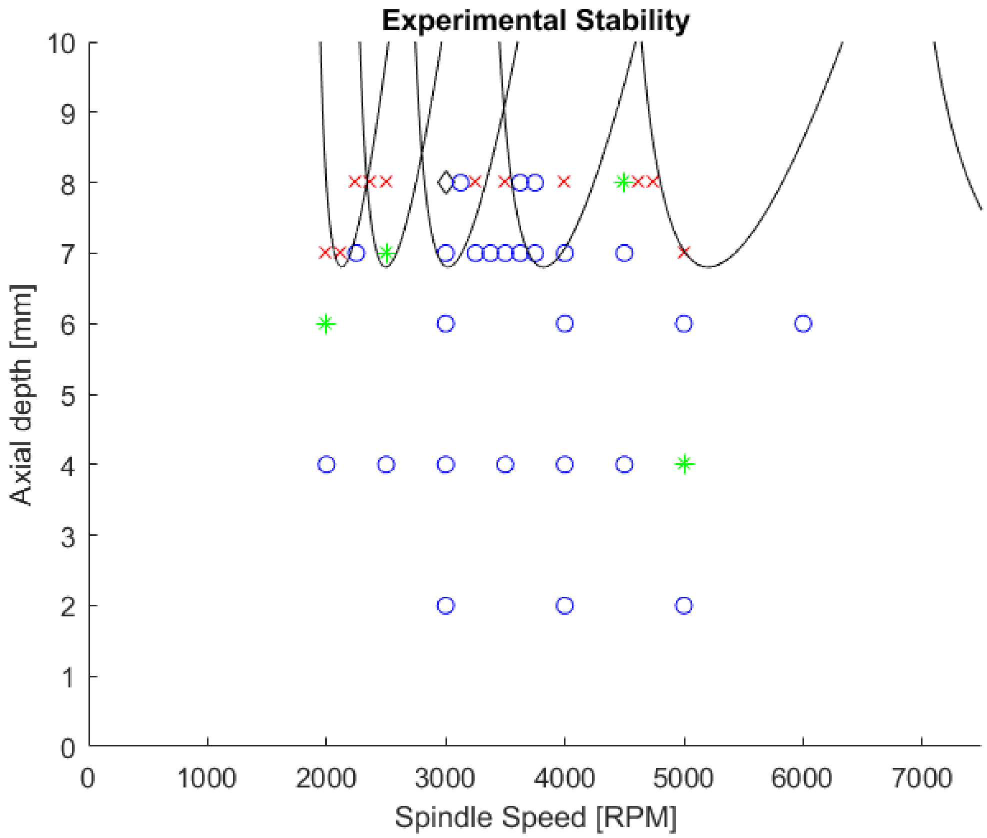


Figure 44: Experimental chatter results based on algorithm analysis of experimental data. Blue circle = stable, red x = chatter, green star = borderline, and black diamond = invalid data. Also included is the theoretically determined lobes, plotted as black lines.

The plots above also contain the theoretical lobes as determined by the average angle approach described in Section 2.4. These lobes are not expected to match perfectly, but to give a sense of the transition regions between stable and unstable cuts. The theoretical model requires a number of simplifying assumptions which reduce its accuracy, particularly in the location of the stability lobes. The axial depth limit, b_{lim}, can still be fairly accurate with the theoretical model.

Stability information from both observation and the algorithm is condensed into a single plot, shown in Figure 45. This plot uses symbols to represent the observed state of the cut, and color for the algorithm's determined state. Also included is the theoretically determined lobes for reference.

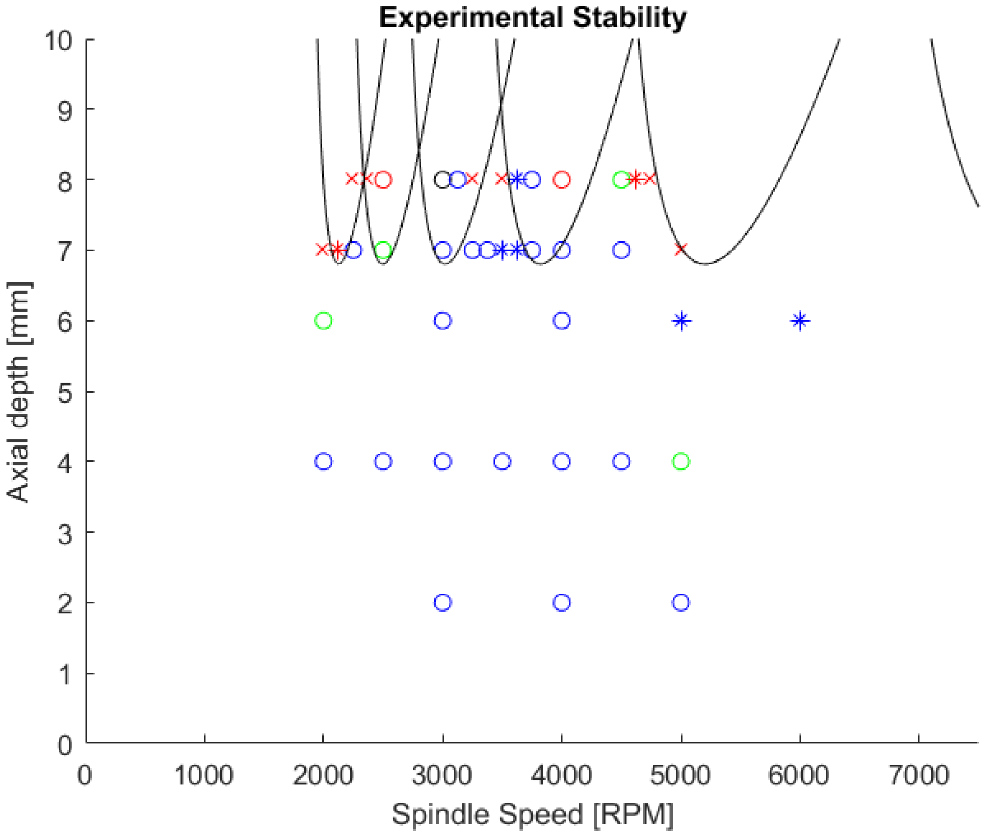


*Figure 45: Experimental chatter results. Symbols relates to observed status of cut (o = stable, x = chatter, * = borderline), and color relates to algorithm results of experimental data (blue = stable, red = chatter, green = borderline, black = unusable data).*

The data collected in the experiments cover a range of behaviors, with several strongly showing the existence of chatter. The observation and the chatter detection algorithm show agreement on almost all the data sets that are clearly stable or unstable. Differences between the two occur in the transitional zone, where the vagueness of the definition of chatter can lead to different interpretations of the cut stability. The fact that even experts come to different conclusions about stability, it is reasonable to expect to the algorithm to not be perfect. Given the examples discussed above, it can be concluded that the algorithm is reasonably good at determining stability given good tuning values, and is a viable solution.

5.4: Comparison of experimental and simulation stability

The experimental data from cuts in Table 5 can also be used to evaluate the accuracy of the simulation program in determining the transition between stable and unstable cuts. Specifically, the stability lobe diagrams are compared for accuracy and trends.

Figure 46 is the simulation results from Section 4.7, and Figure 47 is the experimental results from Section 5.3. Both plots are for quarter immersion up-milling on the Fadal CNC machine. The two plots have a noticeably large difference in the axial depth at which chatter occurs. The simulation transition is between 4 and 5 mm, while the experimental transition is around 7 mm. The simulation points better match the shape of the theoretical lobes, while the experimental points better match the axial depth limit.

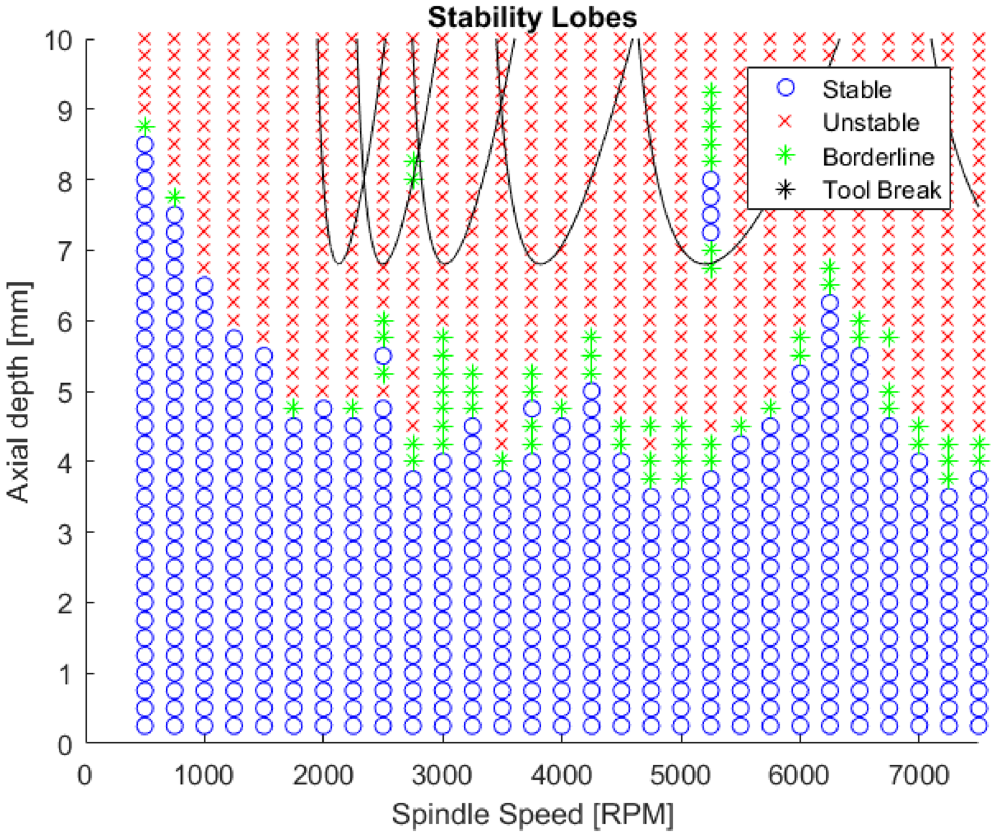


Figure 46: Simulation results originally presented in Section 4.5. Average angle approach for theory.

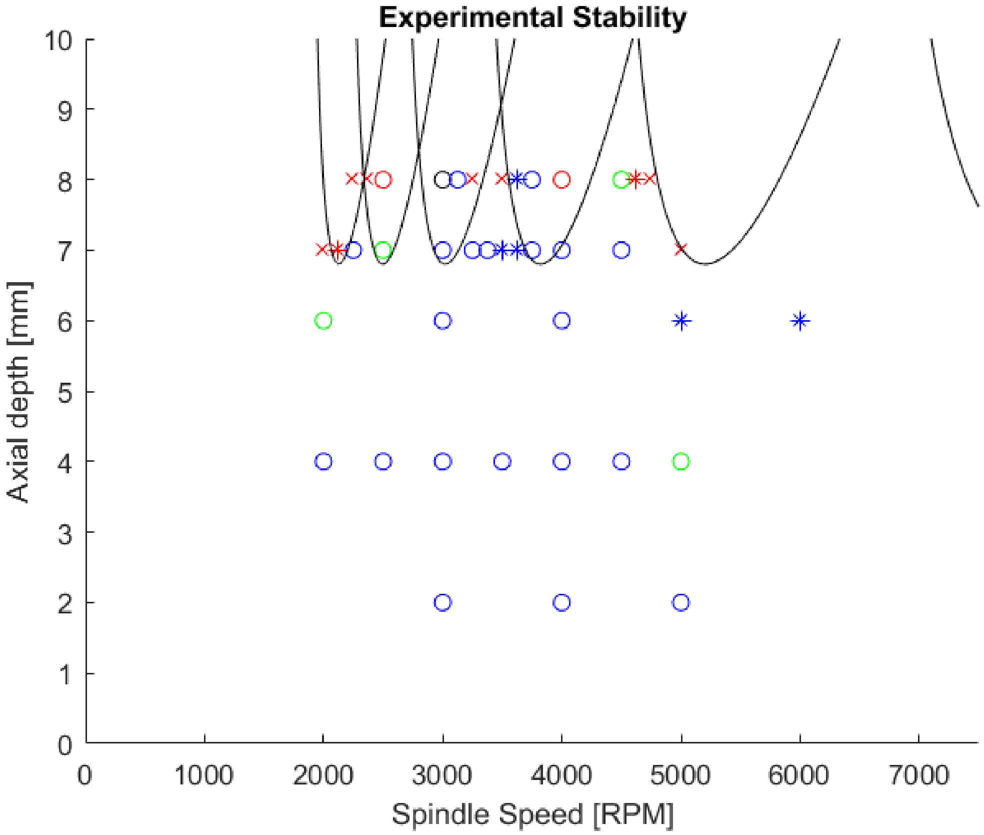


*Figure 47: Experimental chatter results from in Section 5.4. Symbols relates to observed status of cut (o = stable, x = chatter, * = borderline), and color relates to algorithm results of experimental data (blue = stable, red = chatter, green = borderline, black = unusable data).*

There are several reasons for inaccuracy in the simulation chatter prediction. The most obvious is the machining center dynamic model parameters. If k, ω_n, or ζ are not accurate, there is little hope of obtaining accurate chatter prediction. The next chapter explores the effects of these parameters on the stability lobe plots. Another reason is the simplified single mode dynamic model that is used to characterize a very complex machining center. Multiple modes with time varying system parameters is needed to obtain better chatter predictions.

5.5: Summary

This chapter describes the experimental cuts that were made to validate the chatter detection algorithm. These cuts were completed on a Fadal EMC 3-axis milling machine, using a 3-tooth flat end mill, equipped with carbide inserts, and a 6061-aluminum workpiece. The cuts are all up milling with a quarter tool immersion.

Force experienced during the cutting process was measured using a Kistler Dynamometer. The collected force data was then used for simulation verification (presented in Section 3.6), and as the input to the chatter detection algorithm for confirmation of operation.

Several examples of individual cuts are presented to explain the determination of cutting stability by observation. Both stable and unstable cuts are presented, showing the agreement between observation and algorithm. Two additional cases are presented where the stability determination between observation and the algorithm don't agree. The first case shows an example the cutting operation creating a rough surface even though the algorithm states its stable, which is because the large force changes happen over a longer period. The second shows a smooth workpiece surface for an algorithm determined marginal stability, which is because of small force jumps present in the data.

Three plots are presented as visualization for what spindle speed and axial depth combinations generate chatter. The first plot shows the stability determinations made through human observations (i.e. noise generation and surface roughness) of the cut, while plot two shows determinations made by the algorithm. The last plot combines the information from the other two plots to obtain a single visual for easier comparison.

Lastly, the combination experimental stability plot is presented next to the simulation generated stability plot. Both these plots also contain the stability lobes calculated by the average angle approach. These plots show a fair amount of disagreement between each other, mostly in the axial depth limit. The simulation points better match the shape of the theoretical lobes, but the experimental points better match the axial depth limit.

Chapter 6: Dynamic Model Parameter Effects on Stability

6.1: Introduction

This chapter explores the effect of parameter variation on the accuracy of the simulation model to better understand how the model can be improved. Specifically, the spring constant (k) and damping ratio (ζ) of the model are varied to see chatter sensitivity to these changes.

An accurate simulation model of the milling process could be used to select cutting conditions for a given part that avoids chatter. Typically, cutting conditions are chosen using a conservative b_{lim} (axial depth of cut) from stability theory that eliminates the possibility of chatter. Simulation selected axial depth of cuts can be found that are larger than those from theory by slightly adjusting the spindle speed to stability spaces between lobes. This could improve process time while still preventing chatter.

6.2: Effect of spring constant (k)

The first parameter analyzed is the spring constant (k) of the system. This parameter is dependent on the tool, tool-holder, and spindle. Of the portions, the tool is the one part that could be easily redesigned in order to alter the spring constant of the milling system. A close approximation to a cutting tool is a rod, whose stiffness decreases as the rod gets thinner and/or longer.

Three stability lobe plots are generated by simulation, each for a different value of k. All the plots use cutting coefficients of K_{tc} = 600 N/mm^2, K_{rc} = 120 N/mm^2, K_{te} = 20 N/mm, and K_{re} = 19 N/mm, and system parameters of $\omega_{nx} = \omega_{ny}$ = 641 Hz, and $\zeta_x = \zeta_y$ = .131. Figure 48 uses the system k values (k_x = 3.258e6 N/m and k_y = 3.327e6 N/m) found through experiments (same as Figure 32). Figure 49 is generated using k = 2.5e6 N-m, which is approximately 25% smaller than the original, while Figure 50 has k = 4e6 N-m, which is approximately 20% larger. Also, because of the interdependency of variables, changes to k also changes the effective mass (m) and the damping coefficient (c).

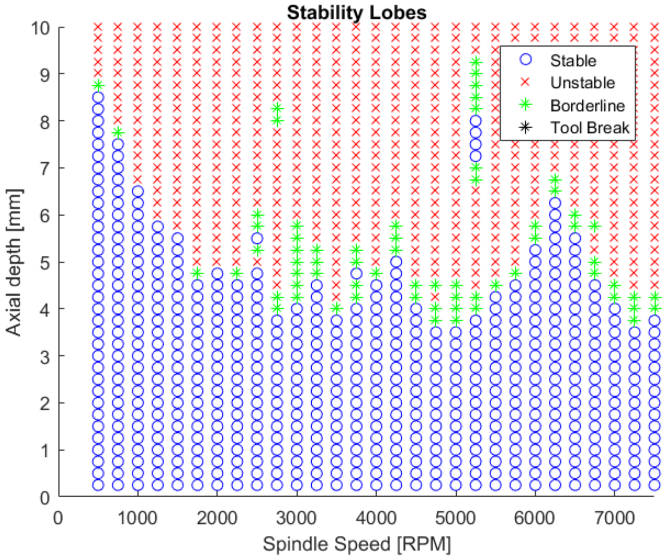


Figure 48: Stability plot for k_x = 3.258e6 N/m and k_y = 3.327e6 N/m, experimental (same as Figure 32).

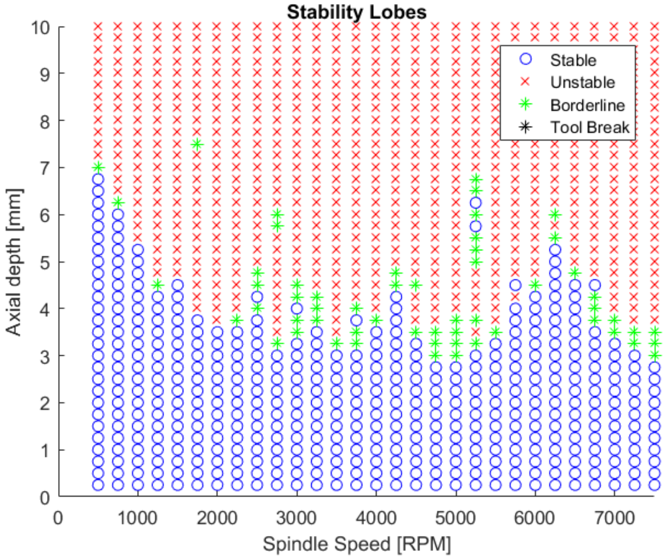


Figure 49: Stability plot for k_x =k_y = 2.5e6 N-m, 25% smaller than experimental (other conditions same as Figure 48).

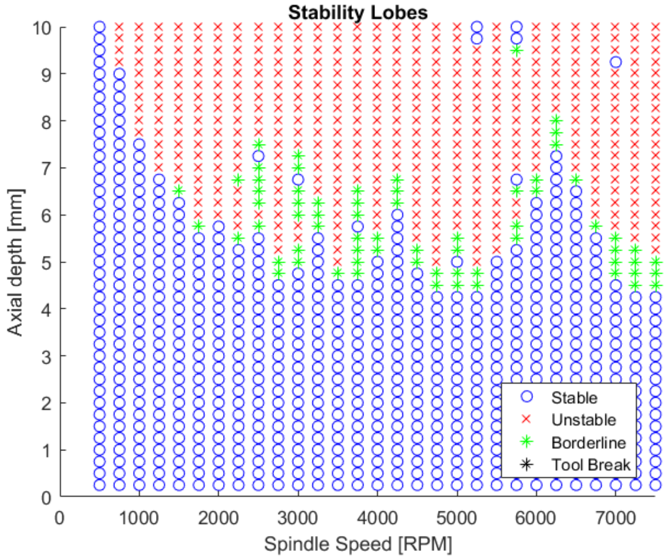


Figure 50: Stability plot for k_x =k_y = 4e6 N-m, 20% larger than experimental (other conditions same as Figure 48).

Comparing the three plots, the most obvious effect observed is the shift in b_{lim}. It appears that the value of k is directly proportional to the value of b_{lim}, i.e. a 20% increase in k shifts the lobes up 20%, and vis-versa. Beside the b_{lim} change, k does not seem to shift the spindle speed at which the peak stable cuts occur, nor change the width or shape of the lobes.

As stated in Section 5.4, the simulation is accurate enough to produce unstable cutting behaviors, but they do not exactly match the experimental results, particularly in b_{lim}. Since increasing k would shift b_{lim} closer to the experimental value, this hints toward the use of a lower k than actually exists in the machine. This is possible since the k used was found through static testing, but the high-speed spinning could make the “observed stiffness” higher.

The simulation results also indicate that use of a stiffer tool could increase the axial depth limit, b_{lim}, allowing deeper cuts to be made, potentially improving process efficiency.

6.3: Effect of damping ratio (ζ)

The effect of the damping ratio (ζ) on the system behavior is explored in this section. Like with the spring constant, this parameter is dependent on the material and shape of the tool edge, and can also depend on the cutting geometry. If the flank of the tool edge rubs on the workpiece the damping can increase substantially, but for only a portion of the tool rotation.

Three different values of ζ are explored; 0.1, 0.131 (nominal), and 0.2. All the simulations use the same nominal conditions as described in Section 6.2, with Figure 51 being the stability plot for the nominal conditions (same as Figure 48). The other two plots use the nominal values, except for ζ. Figure 52 is generated using ζ = .1, which is approximately 24% smaller than the original, while Figure 53 has ζ = .2, which is approximately 53% larger. Also, because of the interdependency of variables, changes to ζ also changes the damping coefficient (c).

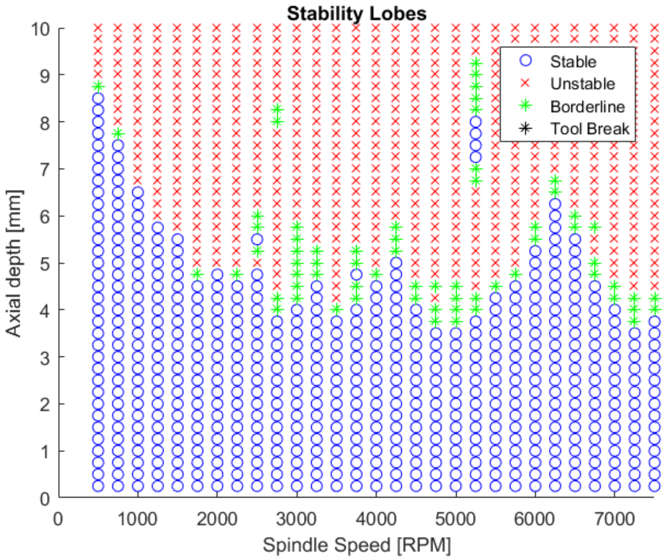


Figure 51: Stability plot for $\zeta_x = \zeta_y = .131$, experimental (same as Figure 32).

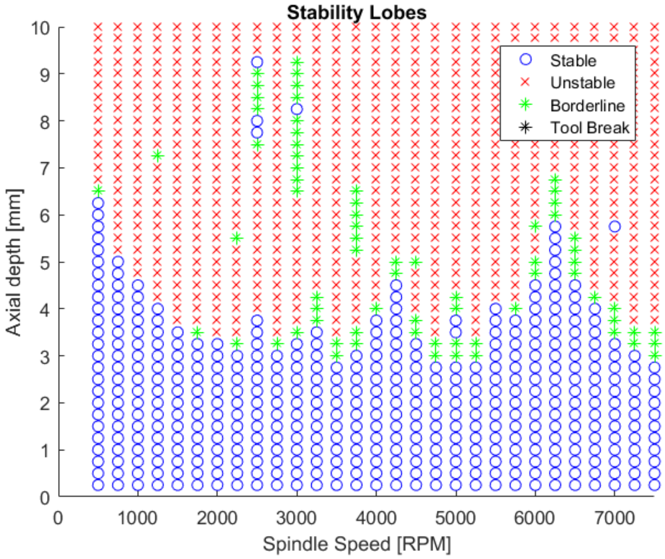


Figure 52: Stability plot for ζ_x = ζ_y = .1, 24% smaller than experimental (other conditions same as Figure 51).

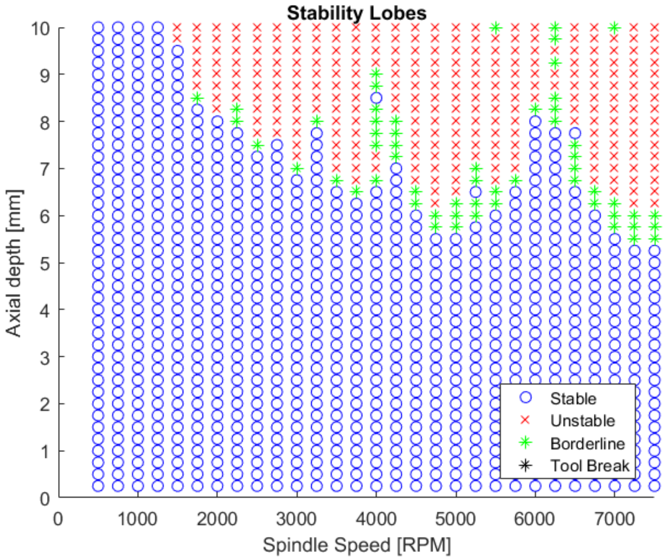


Figure 53: Stability plot for $\zeta_x = \zeta_y = .2$, 53% larger than experimental (other conditions same as Figure 51Figure 48).

Comparing the three plots, the most obvious effect observed, like with k, is the shift in b_{lim}. It appears that the value of ζ is directly proportional to the value of b_{lim}, i.e. a 20% increase in ζ shifts the lobes up 20%, and vis-versa. There is no noticeable spindle speed shifting of the stable peaks caused by ζ, but the shapes of the lobes change slightly. As ζ is increased, the tip of the lobes round out. Also, as ζ increases the low spindle speed cuts have a larger upward shift in axial depth limit than the higher spindle speeds.

While the simulation is accurate enough to produce unstable cutting behaviors, they do not exactly match the experimental observations. Figure 53 indicates that increasing ζ would shift b_{lim} closer to the experimentally determined value, this suggests that the ζ used in the simulation is possibly low. Since the free vibration method used to find the ζ does not consider tool edge and workpiece contact, it is quite possible the value is not accurate. Also, the simplicity of the model used does not consider the additional damping present at higher spindle speeds, and the interdependency of ζ on tool edge geometry and tool wear.

Based on these results, it would seem to be advantageous to cut with a higher damping ratio, since it will allow for the machine to safely operate at higher axial depths. Practically this is difficult to do since it requires more tool/flank face interface which results in high tool wear rate, or a specially designed tool holder that is expensive and less stiff than ordinary tools.

Four more simulations are run where k and ζ are altered simultaneously in order to determine parameter interdependences. Although not shown in the thesis, the plots show the same shifting as described above, with changes in both k and ζ shifting b_{lim} up or down, and ζ creating a larger shift in the lower spindle speed cuts. The new observation is that the amount of shifting caused by changes in k is dependent of the value of ζ. Changes in k at lower ζ values has a smaller effect on the axial depth limit than changes in k at larger ζ values.

6.4: Summary

This chapter explores the effects that the spring constant (k) and damping ratio (ζ) have on the simulation based stability plots. The plots are generated by applying the chatter detection algorithm to simulation force data. Three plots are created for spring constant changes and three for damping ratio changes. The first uses experimentally determined parameters for the Fadal milling machine. The second is with k (or ζ) increased, and the third with k (or ζ) decreased.

In general, it appears that k and ζ are directly proportional to the axial depth limit (b_{lim}) of the milling machine. ζ additionally causes a larger shift at lower spindle speeds. This is complicated by changing k and ζ simultaneously, which shows interdependency between the two parameters. This relation shows larger shifting in association to changes in k when ζ is larger.

Chapter 7: Conclusions and Future Work

7.1: Introduction

This chapter summarizes the work presented in this thesis, along with the major results. Future work is also summarized.

7.2: Conclusions

This research explores the stability of end milling through simulation and experiments. Specifically, an end milling simulation model is revised to more accurately simulate end milling chatter, and a new chatter detection algorithm is created to quickly process cutting force data and identify the existence of chatter. Accurate simulation and on-line detection of chatter will improve process planning and machine operation and make milling more efficient and cost effective. The conclusions of this research are described below.

The simulation uses a linearized, coefficient based model to calculate cutting forces experienced during operation, and applies Eulerian integration to the dynamic equations. This simulation has been used in previous research with good success, but required a few modifications to operate more efficiently. The two improvements made are changing the simulation from revolution-step dependent to time-step dependent, and adding a check that terminates the program if cutting forces exceed 1.9 Mega-Newtons, which will break a ¾” (19.1 mm) tool. The step change increases the accuracy of simulations at low spindle speeds, mostly through lowering the possibility of aliasing. The break check ensures the simulation does not continue running when instabilities cause cutting forces to grow exponentially.

The chatter detection algorithm developed in this research is based on the once per revolution sampling method developed by Schmitz. This method determines the presence of chatter based on microphone data sampled once every revolution of the spindle. Improvements made to this algorithm include using force data instead of sound or displacement data, and processing the once per rev sampling by taking difference between the current force sample and the previous sample in both x and y directions, and calculating the magnitude of the differences. Since steady forces equate to stable cutting, analyzing the spread of changes will show the presence of chatter. The one drawback is that the algorithm requires tuning, which requires looking at a collection of cuts, analyzing them with the algorithm, and choosing reasonable variance values.

The experimental cuts made during this research provided accurate data for both stable and unstable milling operations. Comparing this data to simulation results confirms that the simulation can accurately simulate stable milling operation, but that it tends to show chatter at much lower axial depths for unstable cases. The chatter detection algorithm is applied to the experimental data with good results. Clearly stable and unstable cuts, as determined by observation, are correctly determined by the algorithm. However, marginally stable cuts are more difficult to identify with the algorithm. Stability plots were developed by theory and simulation and compared to the experimental. The axial depth of cut limit was comparable between the theory and experiments, but much lower for the simulation.

The simulation is used to explore chatter's sensitivity to CNC machine dynamic system parameters. By increasing and decreasing the spring constant (k) and damping ratio (ζ), it is observed that the axial depth limit is directly proportional to the change in the parameters.

7.3: Future Work

Work is available to further the work completed during this research. The final goal for this research is a control system that can maintain stable cutting operation of a CNC milling machine. This controller will automatically adjust the feed rate and spindle speed based on determination of cutting stability provided by the chatter detection algorithm.

The research presented in this thesis demonstrates the viability of the chatter detection algorithm for a limited number of cuts. More work is needed to test the algorithm with a variety of tools and workpiece materials. In addition, a consistent and accurate method of variance tuning of the algorithm must be automated for online testing on a CNC machine. The Fadal CNC with an open-architecture controller would be an ideal platform for testing.

It would also be useful to continue improvements to the milling simulation model. An accurate model could be very useful in process planning, e.g. to avoid chatter regions, and in evaluating process control and monitoring algorithms. More work is needed in evaluating the number of machine modes needed to improve simulation accuracy. Also, more reliable methods of determining the modes are needed. Finally, many more experimental cuts are needed to validate the simulation model.

The proposed process monitoring and control system could be improved by designing the chatter controller to work in conjunction with the tool wear system already developed. This would enable the machine to operate closer to the unstable regions of the stability lobe plots, thereby increasing productivity.

References

[1] M. Nouri, "Real-Time Process Monitoring and Control in Milling," University of New Hampshire, Durham, 2015.

[2] T. L. Schmitz, "Catter recongnition by a statistical evaluation of the synchronously sampled audio signal," *Journal of Sound and Vibration,* no. 262, pp. 721-730, 2003.

[3] Y. Altintas and P. K. Chan, "In-Process Detection and Suppression of Chatter in Milling," *International Journal of Machining Tools an Manufacturing,* vol. 32, no. 3, pp. 329-347, 1992.

[4] L. Ma, S. N. Melkote and J. B. Castle, "A Model-Based Computationally Efficient Method of On-Line Detection of Chatter in Milling," *Journal of Manufacturing Science and Engineering,* vol. 135, pp. 1-11, 2013.

[5] Y. Altintas, Manufacturing Automation: Metal Cutting Mechanics, Machine Tool Vibrations, and CNC Design, New York: Cambridge University Press, 2000.

[6] T. L. Schmitz and K. S. Smith, Machining Dynamics: Frequency Respoinse to Improved Productivity, New York: Springer, 2009.

[7] F. Eren, "The Effect of Cutting Force Model Coefficient Variability on Process Planning in Milling," University of New Hampshire, Durham, 2011.

[8] G. Tlusty, Manufacturing Processes and Equipment, Upper Saddle River: Prentice Hall, 2000.

[9] K. Ogata, System Dynamics, Upper Saddle River: Pearson Prentice Hall, 2004, p. 395.

[10] M. H. Koh, "Identification of System Parameters for End Milling Force Simulation with Tool and Wrokpiece Compliance," University of New Hampshire, Durham, 2012.

[11] M. A. Rubeo and T. L. Schmitz, "Global Stability Predictions for Flexible Workpiece Milling using Time Domain Simulation," [Online].

[12] Y. Altintas, G. Stepan, D. Merdol and Z. Dombovari, "Chatter stability of milling in frequency and discrete time domain," *CIRP Journal of Manufacturing Science and Technology,* pp. 35-44, 2008.

[13] E. Budak and Y. Altintas, "Analytical Prediction of Chatter Stability in Milling - Part I: General Formulation," *Journal of Dynamic Systems, Measurement, and Control,* vol. 120, pp. 22-30, 1998.

[14] A. J. Harmon, "Calibration and Characterization of a Low-cost Wireless Sensor for Applications in CNC End Milling," University of New Hampshire, Durham, 2012.

[15] M. Xu, "Smart Machining System Platform for CNC Milling with the Integration of a Power Sensor and Cutting Model," University of New Hampshire, Durham, 2007.

[16] Kistler Group, "Data Sheet, Type 9257B - Kistler," [Online]. Available: https://www.kistler.com/?type=669&fid=51226. [Accessed 2015].

Appendix A: Cutting Simulation Code

```
%% Milling cutting simulation
% Based on code by Schmitz and Smith

% function [t, F, Fx, Fy, x, y, RevStep] = ForceSimulation(rpm, AxialDepth,
RevDes, tmin)

%% Inputs needed to run program independently
clear
clc
rpm = 4250;                 % Spindle Speed (rpm)
AxialDepth = .00475;%.125 * 0.0254;         % (m)
RevDes = 25;                % Desired # Revs to analyze
tmin = .5;

%% ForceSimulition parameter setup
NT = 3;                         % Number of teeth
% Feedrate = .798e-3;    % (m/min)
% FeedperTooth = Feedrate/(rpm*NT); % (m/tooth)

avgChip = .002 * .0254; %m
%   *** Look below for this code ***
% FeedperTooth = avgChip*(StartAngle-EndAngle)*(pi/180)/...
%     (cosd(EndAngle)-cosd(StartAngle)); % (m/tooth)

% FeedperTooth = .15e-3; % (m/tooth)

%% Setup Cutting Conditions
% Tool
ToolDiameter = 0.75*0.0254;     % (m)
HelixAngle = 17.869;            % (deg)
RO = zeros(1,NT)*1e-6;          % (m) tooth-to-tooth runout
% RO = [5 -25 0]*1e-6;

% Cutting Conditions
RadialDepth = ToolDiameter * (.25);  % (m)
CuttingType = 2;                    % 1:Down Milling, 2:Up Milling

% Cutting Coefficients
% Ks = 600e6;     % N/m^2
% beta = 60;      % deg
% Ktc = Ks*sin(beta*pi/180);      % (N/m^2)
% Krc = Ks*cos(beta*pi/180);      % (N/m^2)
% Kte = 0;     % (N/m)
% Kre = 0;      %  (N/m)

% Steel
% Ktc = 1738e6;    %  (N/m^2)
% Krc = 584e6;     %  (N/m^2)
% Kte = 62e3;      %  (N/m)
% Kre = 87e3;      %  (N/m)

% Aluminum
Ktc = 600e6;     %  (N/m^2)
Krc = 120e6;     %  (N/m^2)
Kte = 20e3;      %  (N/m)
Kre = 19e3;      %  (N/m)

% Aluminum (Koh)
% Ktc = 688.0e6;     %  (N/m^2)
% Krc = 229.4e6;     %  (N/m^2)
% Kte = 17.2e3;      %  (N/m)
% Kre = 10.5e3;      %  (N/m)

% Aluminum (Zhao) Valid for .002 to .004 in.
% Ktc = 820e6;     %  (N/m^2)
% Krc = 280e6;     %  (N/m^2)
% Kte = 19e3;      %  (N/m)
% Kre = 12e3;      %  (N/m)

% Time vector
TimeStepDes = 1e-5;     % Desired time step
RevStep = floor(60/(rpm*TimeStepDes));  % Number of steps per rev
TimeStep = 60/(rpm*RevStep);     % Calculate new time step [sec]
TimeEnd = tmin + (RevDes*RevStep*TimeStep);
t = 0:TimeStep:TimeEnd;

dphi = 360/RevStep;      % Angular increment [deg]

% Variables
ToolRadius  = ToolDiameter/2;       %[m]
LagAngle = asin((AxialDepth*tan(HelixAngle*pi/180))/ToolRadius)*180/pi;

% Calculation Start & End Angle depends on cutting type and radial depth
% For slot cutting, it does not matter whether type 1 or 2.
% first quadrant: +X & +Y, clockwise rotation
if CuttingType == 1
    StartAngle = 180 - acos( (ToolRadius-RadialDepth)/ToolRadius )*180/pi;
    EndAngle = 180;
elseif CuttingType == 2
```

```
    StartAngle = 0;
    EndAngle = acos( (ToolRadius-RadialDepth)/ToolRadius )*180/pi;
else
    disp('---Wrong Cutting Type---')
    return
end
ActualEndAngle = EndAngle + LagAngle;

FeedperTooth = -avgChip *(EndAngle-StartAngle)*(pi/180)/...
    (cosd(EndAngle)-cosd(StartAngle)); % (m/tooth)

%% Model start up
% Define modal parameters for x direction
% kx = 8e6;                   % N/m
% zetax = .02;
% wnx = 500*2*pi;             % rad/s
% kx = [2e7 1.5e7];                   % N/m
% zetax = [0.05 0.03];
% wnx = [800 1000]*2*pi;              % rad/s

%smart tool
% kx = [1.21e6 2.02e6];               % N/m
% zetax = [.099 .043];
% wnx = [628 1104]*2*pi;              % rad/s

%Smart tool single mode
kx = 3.258e6;               % N/m
zetax = .131;
wnx = 641*2*pi;             % rad/s
mx = kx./(wnx.^2);                  % kg
cx = 2*zetax.*(mx.*kx).^0.5;        % N-s/m
x_modes = length(kx);               % number of modes

% Define modal parameters for y direction
% ky = 8e6;                   % N/m
% zetay = .02;
% wny = 500*2*pi;             % rad/s
% ky = [2e7 1.5e7];                   % N/m
% zetay = [0.05 0.03];
% wny = [800 1000]*2*pi;              % rad/s

%smart tool
% ky = [1.22e6 2.62e6];                  % N/m
% zetay = [.110 .041];
% wny = [620 1328]*2*pi;                 % rad/s
```

```
%Smart tool single mode
ky = 3.327e6;                % N/m
zetay = .131;
wny = 641*2*pi;             % rad/s
my = ky./(wny.^2);                  % kg
cy = 2*zetay.*(my.*ky).^0.5;        % N-s/m
y_modes = length(ky);               % number of modes

%% Interate model through time
for i = 1:length(t)
if t(i) == 0        % Time = 0, Initial values set
    phi = 0:dphi:(360-dphi);
    p = zeros(x_modes,length(t));
    dp = zeros(x_modes,length(t));
    q = zeros(x_modes,length(t));
    dq = zeros(x_modes,length(t));
    x = zeros(1,length(t));
    y = zeros(1,length(t));
    if HelixAngle == 0
        da = AxialDepth;
    else
        % discretized axial depth [m]
        da = ToolDiameter*(dphi*pi/180)/( 2*tan(HelixAngle*pi/180) );
    end
    AxialStep = round(AxialDepth/da);
    surf = zeros(AxialStep,RevStep);
    Fx = zeros(1,length(t));
    Fy = zeros(1,length(t));
    F = zeros(1,length(t));
    teeth = zeros(1,4);
    if NT>1
        for cnt = 2:NT
            teeth(cnt) = teeth(cnt-1)+floor(RevStep/NT);%ceil(RevStep/4);
            if teeth(cnt)>RevStep
                teeth(cnt) = teeth(cnt)-RevStep;
            end
        end
    end

else                % Time > 0, Iterate through time
    %%%%%%%%%%%%%%%%%%%%%%%% Simulation %%%%%%%%%%%%%%%%%%%%%%%%
    for cnt2 = 1:NT     % current teeth positions
        teeth(cnt2) = teeth(cnt2) + 1;
        if teeth(cnt2) > RevStep
            teeth(cnt2) = teeth(cnt2)-RevStep;
        end
```

```
end

for cnt3 = 1:NT     % cycle through all teeth
    for cnt4 = 1:AxialStep  %cycle through entire height
        cntphi = teeth(cnt3) - (cnt4-1); % based on bottom of teeth

        if cntphi < 1 % helix has wrapped through phi = 0 [deg]
            cntphi = cntphi + RevStep;
        end

        phia = phi(cntphi);     % angle for given axial disk [deg]

        % Determine current cutting forces
        if (StartAngle <= phia) && (phia <= ActualEndAngle)
            n = x(i-1) * sin(phia*pi/180) ...
                -y(i-1) * cos(phia*pi/180); % [m]
            dt = FeedperTooth*sin(phia*pi/180) + ...
                surf(cnt4, cntphi) - n + RO(cnt3); % [m]
            if dt < 0
                dFt = 0;
                dFr = 0;
                surf(cnt4, cntphi) = surf(cnt4, cntphi) + ...
                                   FeedperTooth*sin(phia*pi/180);
            else
                dFt = Ktc*dt*da + Kte*da;
                dFr = Krc*dt*da + Kre*da;
                surf(cnt4, cntphi) = n - RO(cnt3);
            end
        else
            dt = 0;
            dFt = 0;
            dFr = 0;
        end

        %%%%%% Force output %%%%%%
        Fx(i) = Fx(i) + dFt*cos(phia*pi/180) + dFr*sin(phia*pi/180);
        Fy(i) = Fy(i) + dFt*sin(phia*pi/180) - dFr*cos(phia*pi/180);

    end
end

F(i) = sqrt(Fx(i)^2 + Fy(i)^2);

if F(i) > 1.9e6
    F = NaN;
    disp('---Tool Broken---')
```

```
        return
    end

    % Numerical integration for position
    % x direction
    for cnt5 = 1:x_modes
        ddp = (Fx(i) - cx(cnt5)*dp(cnt5,i-1) - kx(cnt5)*p(cnt5,i-
1))/mx(cnt5);
        dp(cnt5,i) = dp(cnt5,i-1) + ddp*TimeStep;
        p(cnt5,i) = p(cnt5,i-1) + dp(cnt5,i)*TimeStep;
        x(i) = x(i) + p(cnt5,i);        % m
    end

    % y direction
    for cnt5 = 1:y_modes
        ddq = (Fy(i) - cy(cnt5)*dq(cnt5,i-1) - ky(cnt5)*q(cnt5,i-
1))/my(cnt5);
        dq(cnt5,i) = dq(cnt5,i-1) + ddq*TimeStep;
        q(cnt5,i) = q(cnt5,i-1) + dq(cnt5,i)*TimeStep;
        y(i) = y(i) + q(cnt5,i);        %m
    end

end     % Time if
end     % Time for loop
% end
%%%%%%%%%%%%%%%%%%%%%%%% Function Ends %%%%%%%%%%%%%%%%%%%%%%%%
```

Appendix B: Chatter Detection Algorithm Code

```
%% Code for chatter detection algorithm
% Jonathan Shepard

function [chatter,variance] = chatter_detect(t, F, Fx, Fy, x, y, RevStep, 
method, tmin)
%simulation
if isnan(F) == 1 %Check for tool breaking
    chatter = 3;
    variance = NaN;

% method = 'oneperrev';
elseif strcmp(method,'oneperrev')
    nstart = find(t > tmin,1);
    nstart = find(F == max(F(1:nstart)), 1);
    nsamp = nstart:RevStep:length(t);
    t1 = find(t(nsamp) > tmin, 1);
    t1 = nsamp(t1);

%     xone = x(t1:RevStep:length(t));
% %     xvar = var(xone);
%     yone = y(t1:RevStep:length(t));
% %     yvar = var(yone);
%     dif = zeros(1,length(xone)-1);
%     for i = 1:length(xone)-1
%         dif(i) = sqrt((xone(i)-xone(i+1))^2 + (yone(i)-yone(i+1))^2);
%     end
%     variance = var(dif);

%     mag = (xone.^2 + yone.^2).^.5;
%     variance = var(yone);

    Fxone = Fx(t1:RevStep:length(t));
%     xvar = var(xone);
    Fyone = Fy(t1:RevStep:length(t));
%     yvar = var(yone);
    dif = zeros(1,length(Fxone)-1);
    for i = 1:length(Fxone)-1
        dif(i) = sqrt((Fxone(i)-Fxone(i+1))^2 + (Fyone(i)-Fyone(i+1))^2);
    end
    variance = var(dif);

%     Fone = F(t1:RevStep:length(t));
%     for i = 1:length(Fone)-1
%         dif(i) = sqrt((Fone(i)-Fone(i+1))^2);
%     end
%     variance = var(dif);

%% Limits set by user
    if variance <= 10
        chatter = 0; %Stable
    elseif variance < 5e3 && variance > 10
        chatter = 2; %Borderline
    else
        chatter = 1; %Chatter
    end

else
    disp('---Incorrect Method---')
end % method selection end
end % function end
```

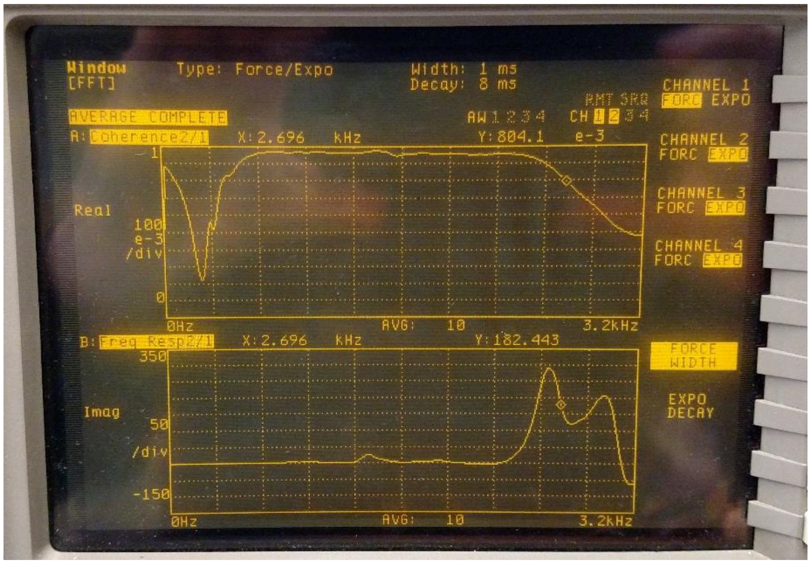
Appendix C: Code for stability lobe generation using simulation

```
%% Stability lobe generation
% Jonathan Shepard

clear
close all
clc

%% Parameters
rpm_start = 500;
rpm_max = 7500;
rpm_step = 250;
axial_start = .25e-3;
axial_max = 10e-3;
axial_step = .25e-3;

method = 'oneperrev';

RevDes = 25;

%% Initialization
figure(1)
hold on
title('Stability Lobes')
xlabel('Spindle Speed [RPM]')
ylabel('Axial depth [mm]')
plot(rpm_max+50,axial_max*10^3+50,'bo')
plot(rpm_max+50,axial_max*10^3+50,'rx')
plot(rpm_max+50,axial_max*10^3+50,'g*')
plot(rpm_max+50,axial_max*10^3+50,'k*')
axis([0 rpm_max 0 axial_max*10^3])
legend('Stable','Unstable','Borderline','Tool Break')
variance = zeros(axial_max/axial_step,rpm_max/rpm_step);

%% Simulation looping
for rpm = rpm_start:rpm_step:rpm_max
    for AxialDepth = axial_start:axial_step:axial_max
        if rpm <= 500
            tmin = 1;
        elseif rpm > 500 && rpm <= 1000
            tmin = .5;
        elseif rpm > 1000
            tmin = .2;
        end
        %Run simulation
        [t,F,Fx,Fy,x,y,RevStep] = ForceSimulation(rpm, AxialDepth, RevDes,
tmin);

        %Check for chatter
        if isnan(F) == 1
            chatter = 3;
        else
           [chatter,variance(floor(AxialDepth/axial_step),rpm/rpm_step)]...
                = chatter_detect(t, F, Fx, Fy, x, y, RevStep, method, tmin);
        end

        %Mark point on plot
        if chatter == 0 %No chatter exists
            plot(rpm,AxialDepth*10^3,'bo')
        elseif chatter == 1 %Chatter exists
            plot(rpm,AxialDepth*10^3,'rx')
        elseif chatter == 2 %Unsure chatter exists
            plot(rpm,AxialDepth*10^3,'g*')
        elseif chatter == 3 %Tool breaks
            plot(rpm,AxialDepth*10^3,'k*')
        end

    end
end

savefig('coolplot')
save('stuff','variance')
```

Appendix D: Tap test Data

Data for Smart tool free vibration/known displacement

X1

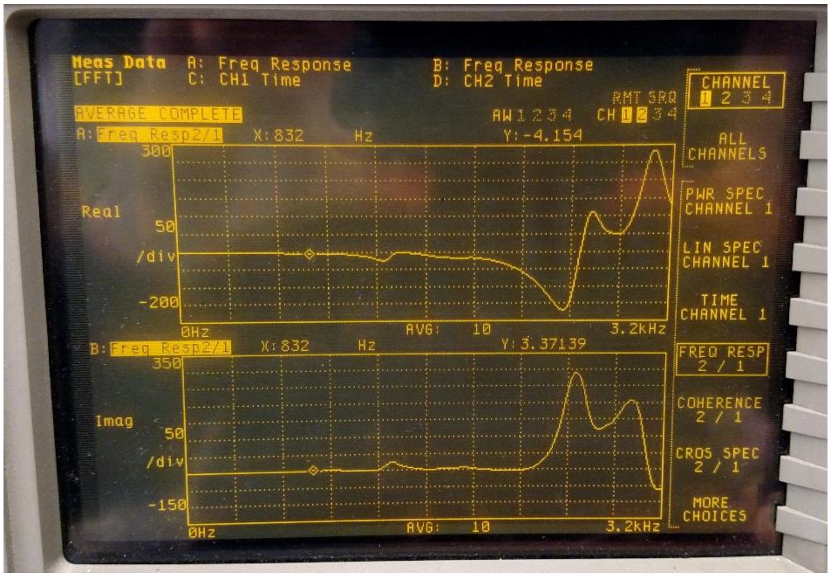


Window [FFT] Type: Force/Expo Width: 1 ms Decay: 8 ms

AVERAGE COMPLETE

A: Coherence2/1 X: 2.696 kHz Y: 804.1 e-3

B: Freq Resp2/1 X: 2.696 kHz Y: 182.443



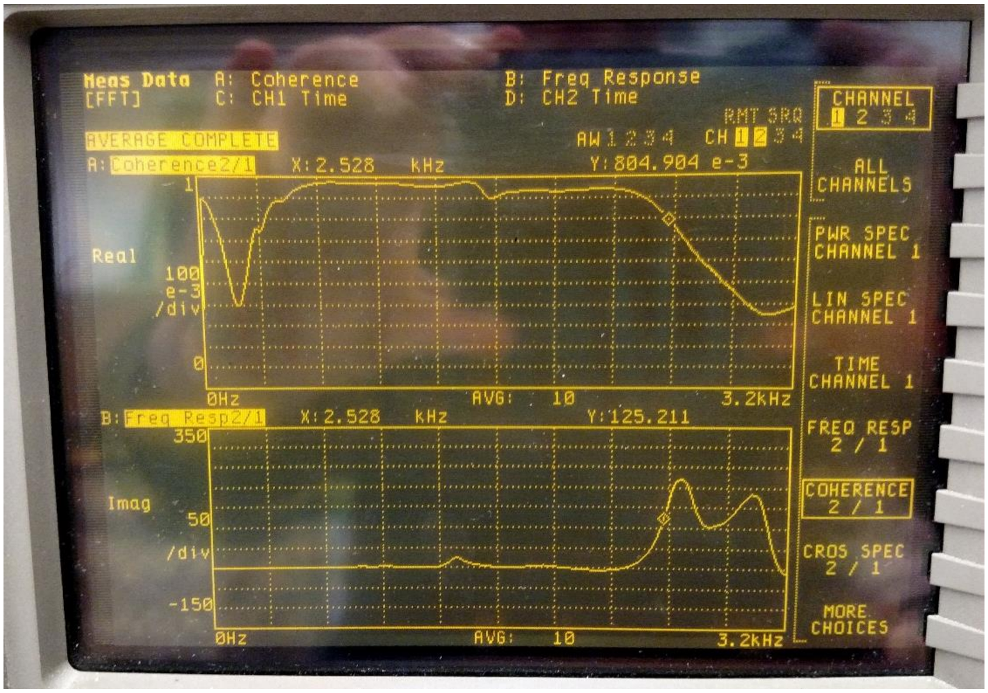
Meas Data [FFT] A: Freq Response C: CH1 Time B: Freq Response D: CH2 Time

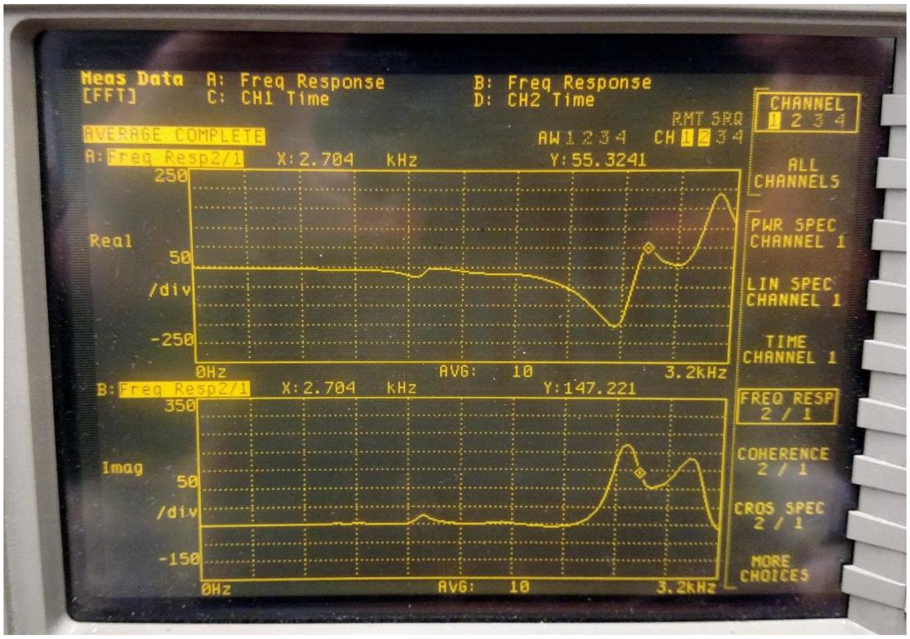
AVERAGE COMPLETE

A: Freq Resp2/1 X: 832 Hz Y: -4.154

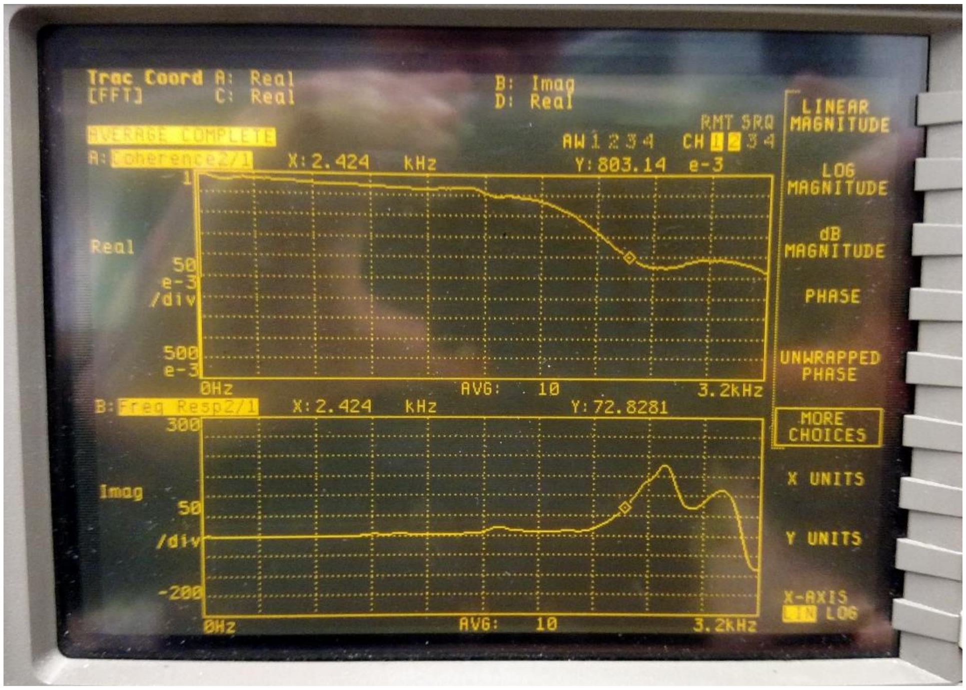
B: Freq Resp2/1 X: 832 Hz Y: 3.37139

X2


Meas Data A: Coherence B: Freq Response
[FFT] C: CH1 Time D: CH2 Time
RMT SRQ
AVERAGE COMPLETE
AW 1 2 3 4 CH 1 2 3 4
A: Coherence2/1 X: 2.528 kHz Y: 804.904 e-3
1
Real
100
e-3
/div
0
0Hz AVG: 10 3.2kHz
B: Freq Resp2/1 X: 2.528 kHz Y: 125.211
350
Imag
50
/div
-150
0Hz AVG: 10 3.2kHz
CHANNEL
1 2 3 4
ALL CHANNELS
PWR SPEC CHANNEL 1
LIN SPEC CHANNEL 1
TIME CHANNEL 1
FREQ RESP 2 / 1
COHERENCE 2 / 1
CROS SPEC 2 / 1
MORE CHOICES


Meas Data A: Freq Response B: Freq Response
[FFT] C: CH1 Time D: CH2 Time
RMT SRQ
AVERAGE COMPLETE
AW 1 2 3 4 CH 1 2 3 4
A: Freq Resp2/1 X: 2.704 kHz Y: 55.3241
250
Real
50
/div
-250
0Hz AVG: 10 3.2kHz
B: Freq Resp2/1 X: 2.704 kHz Y: 147.221
350
Imag
50
/div
-150
0Hz AVG: 10 3.2kHz
CHANNEL
1 2 3 4
ALL CHANNELS
PWR SPEC CHANNEL 1
LIN SPEC CHANNEL 1
TIME CHANNEL 1
FREQ RESP 2 / 1
COHERENCE 2 / 1
CROS SPEC 2 / 1
MORE CHOICES

Y1

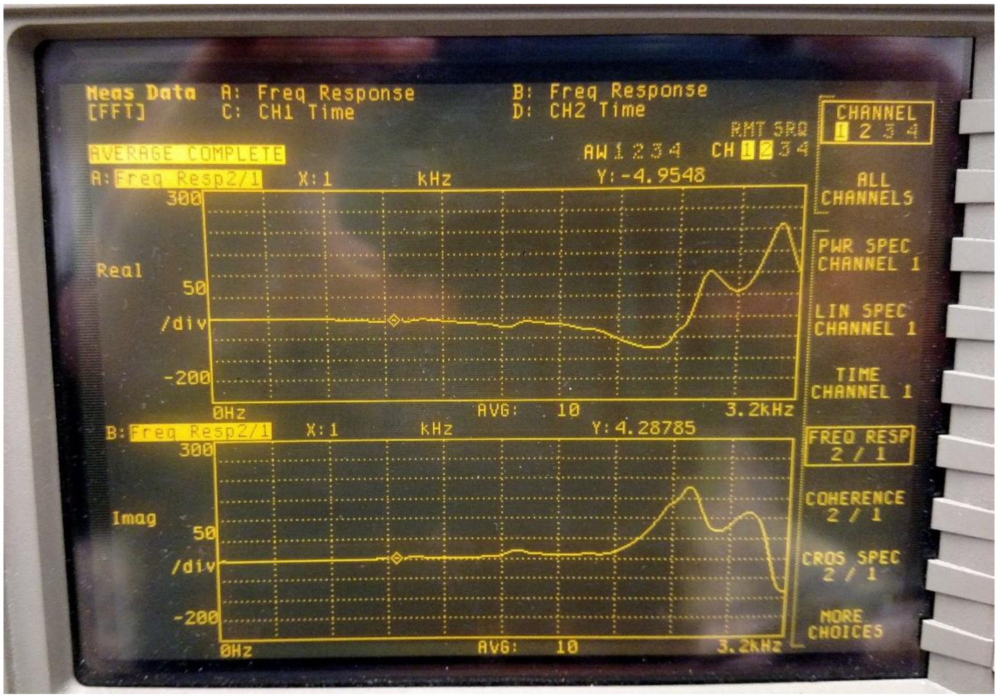


Trac Coord A: Real B: Imag
[FFT] C: Real D: Real

AVERAGE COMPLETE

A: Coherence2/1 X: 2.424 kHz Y: 803.14 e-3

B: Freq Resp2/1 X: 2.424 kHz Y: 72.8281



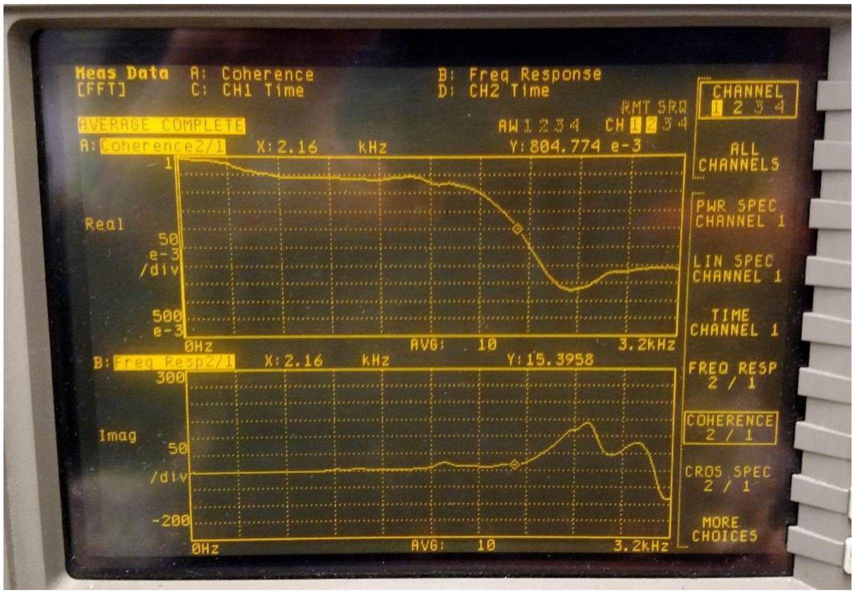
Meas Data A: Freq Response B: Freq Response
[FFT] C: CH1 Time D: CH2 Time

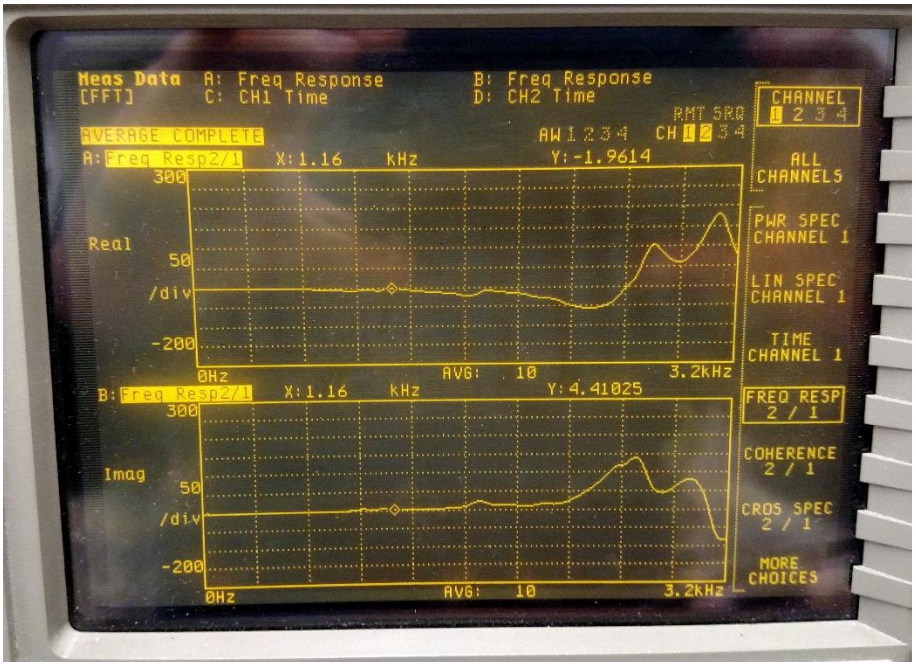
AVERAGE COMPLETE

A: Freq Resp2/1 X: 1 kHz Y: -4.9548

B: Freq Resp2/1 X: 1 kHz Y: 4.28785

Y2


Meas Data [FFT]
A: Coherence
B: Freq Response
C: CH1 Time
D: CH2 Time
RMT SRQ
AVERAGE COMPLETE
AW 1 2 3 4
CH 1 2 3 4
A: Coherence2/1
X: 2.16 kHz
Y: 804.774 e-3
1
Real
50 e-3 /div
500 e-3
0Hz
AVG: 10
3.2kHz
B: Freq Resp2/1
X: 2.16 kHz
Y: 15.3958
300
Imag
50 /div
-200
0Hz
AVG: 10
3.2kHz
CHANNEL 1 2 3 4
ALL CHANNELS
PWR SPEC CHANNEL 1
LIN SPEC CHANNEL 1
TIME CHANNEL 1
FREQ RESP 2 / 1
COHERENCE 2 / 1
CROS SPEC 2 / 1
MORE CHOICES


Meas Data [FFT]
A: Freq Response
B: Freq Response
C: CH1 Time
D: CH2 Time
RMT SRQ
AVERAGE COMPLETE
AW 1 2 3 4
CH 1 2 3 4
A: Freq Resp2/1
X: 1.16 kHz
Y: -1.9614
300
Real
50 /div
-200
0Hz
AVG: 10
3.2kHz
B: Freq Resp2/1
X: 1.16 kHz
Y: 4.41025
300
Imag
50 /div
-200
0Hz
AVG: 10
3.2kHz
CHANNEL 1 2 3 4
ALL CHANNELS
PWR SPEC CHANNEL 1
LIN SPEC CHANNEL 1
TIME CHANNEL 1
FREQ RESP 2 / 1
COHERENCE 2 / 1
CROS SPEC 2 / 1
MORE CHOICES

Data for Smart tool modal fitting

	A	wn	w1	w2	A(convt)	zeta	k	m	c
x1	6.700	624	570	690	-3.880E-05	0.096	1.34E+05	0.34	41.30
	3.171	1104	1056	1152	-5.866E-06	0.043	1.96E+06	1.61	154.42
	20.000	2170	2080	2260	-9.577E-06	0.041	1.26E+06	0.27	48.12
x2	8.076	632	568	696	-4.559E-05	0.101	1.08E+05	0.27	34.71
	3.000	1104	1056	1152	-5.550E-06	0.043	2.07E+06	1.70	163.20
	24.756	2168	2088	2256	-1.188E-05	0.039	1.09E+06	0.23	38.84
x avg	7.388	628	569	693	-4.220E-05	0.099	1.21E+05	0.31	38.00
	3.085	1104	1056	1152	-5.708E-06	0.043	2.02E+06	1.65	158.81
	22.378	2169	2084	2258	-1.073E-05	0.040	1.17E+06	0.25	43.48
y1	6.500	624	560	696	-3.764E-05	0.109	1.22E+05	0.31	42.57
	3.524	1328	1256	1368	-4.506E-06	0.042	2.63E+06	1.49	167.11
	17.412	2264	2192	2344	-7.660E-06	0.034	1.94E+06	0.38	57.66
y2	6.254	616	560	696	-3.717E-05	0.110	1.22E+05	0.32	43.68
	3.836	1328	1264	1368	-4.905E-06	0.039	2.60E+06	1.48	153.53
	15.814	2272	2200	2352	-6.908E-06	0.033	2.16E+06	0.42	63.71
y avg	6.377	620	560	696	-3.740E-05	0.110	1.22E+05	0.32	43.13
	3.680	1328	1260	1368	-4.705E-06	0.041	2.62E+06	1.48	160.32
	16.613	2268	2196	2348	-7.284E-06	0.034	2.05E+06	0.40	60.69