

University of New Hampshire

University of New Hampshire Scholars' Repository

Master's Theses and Capstones

Student Scholarship

Spring 2012

Calibration and characterization of a low-cost wireless sensor for applications in CNC end milling

Andrew James Harmon
University of New Hampshire, Durham

Follow this and additional works at: <https://scholars.unh.edu/thesis>

Recommended Citation

Harmon, Andrew James, "Calibration and characterization of a low-cost wireless sensor for applications in CNC end milling" (2012). *Master's Theses and Capstones*. 704.
<https://scholars.unh.edu/thesis/704>

This Thesis is brought to you for free and open access by the Student Scholarship at University of New Hampshire Scholars' Repository. It has been accepted for inclusion in Master's Theses and Capstones by an authorized administrator of University of New Hampshire Scholars' Repository. For more information, please contact Scholarly.Communication@unh.edu.

**CALIBRATION AND CHARACTERIZATION OF A LOW-COST WIRELESS SENSOR
FOR APPLICATIONS IN CNC END MILLING**

BY

ANDREW JAMES HARMON

B.S., University of New Hampshire, 2011

THESIS

Submitted to the University of New Hampshire

in Partial Fulfillment of

the Requirements for the Degree of

Master of Science

in

Mechanical Engineering

May, 2012

UMI Number: 1518009

All rights reserved

INFORMATION TO ALL USERS

The quality of this reproduction is dependent upon the quality of the copy submitted.

In the unlikely event that the author did not send a complete manuscript and there are missing pages, these will be noted. Also, if material had to be removed, a note will indicate the deletion.

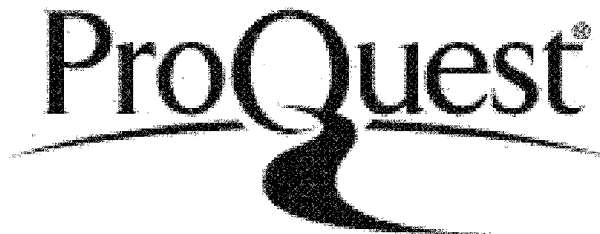


UMI 1518009

Published by ProQuest LLC 2012. Copyright in the Dissertation held by the Author.

Microform Edition © ProQuest LLC.

All rights reserved. This work is protected against unauthorized copying under Title 17, United States Code.



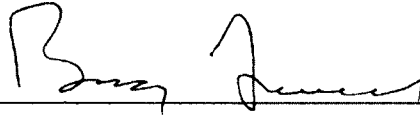
ProQuest LLC
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106-1346

ALL RIGHTS RESERVED

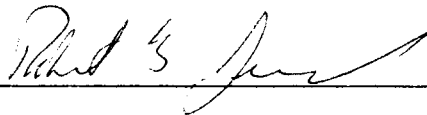
© 2012

Andrew James Harmon

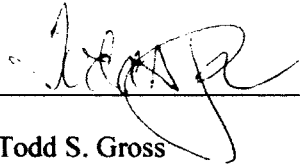
This thesis has been examined and approved.



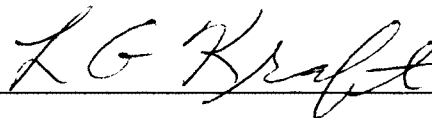
Thesis Director, Dr. Barry K. Fussell
Professor of Mechanical Engineering



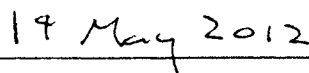
Thesis Co-Director, Dr. Robert B. Jerard
Professor of Mechanical Engineering



Dr. Todd S. Gross
Professor of Mechanical Engineering



Dr. L. Gordon Kraft
Professor Emeritus of Electrical and Computer Engineering



Date

DEDICATION

To mom and dad

ACKNOWLEDGEMENTS

I would like to thank Dr. Barry Fussell and Dr. Robert Jerard for their constant guidance and support. It was a privilege to work with them on this project.

I would also like to thank Christopher Suprock who laid the foundation for this work in his Doctoral Dissertation.

TABLE OF CONTENTS

DEDICATION	iii
ACKNOWLEDGEMENTS	iv
LIST OF TABLES	viii
LIST OF FIGURES	ix
LIST OF SYMBOLS	xv
ABSTRACT	xx

CHAPTER	PAGE
1. INTRODUCTION	1
1.1 Introduction.....	1
1.2 Thesis Overview	2
2. WIRELESS SENSOR DESIGN	4
2.1 Introduction.....	4
2.2 Project Statement	4
2.3 Sensor Design Criteria	5
2.4 Overview of Smart Tool v.10	6
2.5 Strain Gage Selection.....	7
2.6 Orthogonal Decomposition of Strain	7
2.7 Bending Sensitivity Analysis.....	11

2.8	Cross-Sensitivity Analysis.....	15
2.9	Sampling and Data Transmission	19
2.10	Summary.....	20
3.	STATIC CALIBRATION.....	21
3.1	Introduction.....	21
3.2	Calibration of Bending Sensitivity	21
3.3	Calibration of Bending Crosstalk	25
3.4	Calibration of Torsional Crosstalk.....	27
3.5	Calibration of Axial Sensitivity	30
3.6	Characterization of Sensor Noise	32
3.7	Drift and Sensor Stability	33
3.8	Summary.....	34
4.	DYNAMIC CHARACTERIZATION.....	38
4.1	Introduction.....	38
4.2	Experimental Determination of the Static Frequency Response Function	38
4.3	Variation in the Natural Frequency and Damping Ratio with Spindle Speed	42
4.4	Boundary Conditions and Dynamic Parameter Identification	43
4.5	Mass Distribution and Systematic Bias	45
4.6	Summary.....	46
5.	EXPERIMENTAL VALIDATION.....	47
5.1	Introduction.....	47

5.2 Comparison of Kistler and Smart Tool v.10.....	47
5.3 Summary.....	54
6. APPLICATIONS IN SIGNAL PROCESSING.....	55
6.1 Introduction.....	55
6.2 Linear Predictive Coding for Dynamic Parameter Estimation	61
6.3 A Simple Algorithm for Enhanced Chatter Frequency Detection	76
6.4 Model-Based Optimal Filtering.....	79
6.5 Sub-Optimal Filtering for Signal Enhancement	95
6.6 Summary.....	101
7. CONCLUSIONS AND FUTURE WORK.....	103
7.1 Conclusions.....	103
7.2 Extension of Signal Processing Techniques for Real-Time Implementation	106
7.3 Suggested Topics and Direction for Future Work	110
 LIST OF REFERENCES.....	 113
APPENDIX A: EXPERIMENTAL PROTOCOLS FOR STATIC CALIBRATION	115
APPENDIX B: PROTOCOL FOR CONSTRUCTION OF SMART TOOL V.10.....	128
APPENDIX C: DERIVATION OF CROSSTALK THEORY DUE TO BRIDGE MISALIGNMENT	131
APPENDIX D: DATA REDUCTION PROTOCOL FOR EXPERIMENTAL VALIDATION	138
APPENDIX E: ARCHIVE OF MATLAB M-FILES.....	157

LIST OF TABLES

Table 2.1 – Sensor Design Criteria	5
Table 2.2 - Table of design specifications for calculation of bending sensitivity.....	13
Table 3.1 - Summary of bending crosstalk calibration	26
Table 3.2 - Summary of torsional crosstalk experimentation	29
Table 6.1 - Comparison of estimators useful for signal processing.....	102
Table A.1 – Calibration of radial bending sensitivity	121
Table A.2 – Calibration of tangential bending sensitivity	122
Table A.3 – Calibration of radial bending cross-sensitivity	123
Table A.4 – Calibration of tangential bending cross-sensitivity.....	124
Table A.5 – Calibration of torsional cross sensitivity.....	125
Table A.6 – Calibration of axial sensitivity	126
Table A.7 - Table of raw data for drift investigation	127

LIST OF FIGURES

Figure 1.1 - Our Smart Tool sensor, version 10	2
Figure 2.1 - Overview of the data transmission sequence	6
Figure 2.2 - Schematic of a semiconductor Wheatstone bridge purchased from Suprock Technologies.....	7
Figure 2.3 - Positioning of the Wheatstone bridges to isolate bending strains.....	8
Figure 2.4 - Beam model illustrating orthogonal decomposition of bending strain	9
Figure 2.5 - Resolution and span versus instrumentation amplifier gain.....	14
Figure 2.6 - Illustration of misalignment errors considered in this analysis.....	15
Figure 2.7 – Theoretical measurement error on the radial bridge due to bending crosstalk.....	17
Figure 2.8 - Theoretical measurement error on the tangential bridge due to bending crosstalk....	17
Figure 2.9 - Torsional crosstalk as a result of gage misalignment.....	18
Figure 2.10 - Data transmission board designed by Jeffery Nichols [6, 14].....	19
Figure 3.1 – Instron servo-hydraulic machine used to apply bending moments to the sensor	22
Figure 3.2 – Schematic of the load transmission through the sensor using the C-channel calibration fixture.....	22
Figure 3.3 – Lumped parameter schematic of the Instron load train	23
Figure 3.4 – Calibration curve for static bending sensitivity in the tangential direction	24
Figure 3.5 – Calibration curve for static bending sensitivity in the radial direction.....	24
Figure 3.6 - Bending crosstalk results from circumferential misalignment of the Wheatstone bridge	25
Figure 3.7 - Illustration of stress transformation resulting from rotation of the gage element	27
Figure 3.8 - Static calibration test fixture for torsional loading.....	27

Figure 3.9 – Calibration of torsional crosstalk on the tangential bridge.....	28
Figure 3.10 - Calibration of torsional crosstalk on the radial bridge	28
Figure 3.11 - Calibration curve for axial sensitivity of the Smart Tool v.10.....	30
Figure 3.12 - Comparison of the axial force to the in-plane X and Y cutting forces.....	31
Figure 3.13 - Sensor noise is shown to follow a Gaussian distribution.....	32
Figure 3.14 - Study of sensor drift on both bridges for extended time records	33
Figure 3.15 - Static confidence intervals for measured tangential force.....	36
Figure 3.16 - Static confidence intervals for measured radial force	36
Figure 4.1 - Schematic of the hammer test experimental setup.....	39
Figure 4.2 - Force input and accelerometer response measured by the spectrum analyzer	40
Figure 4.3 - Estimated frequency response function and squared coherency spectrum	40
Figure 4.4 - Estimated FRF and corresponding FRF curve fit.....	41
Figure 4.5 - Dynamic variation in the natural frequency and damping ratio.....	42
Figure 4.6 - Example of using LPC for system identification; the LPC-based PSD estimate is formed from the auto-regressive model to illustrate the frequency response of the system model	44
Figure 4.7 - Systematic bias as a result of uneven mass distribution around the tool holder	45
Figure 5.1 - Flow chart of data processing required for net force comparison.....	49
Figure 5.2 - Comparison of net force profiles between Smart Tool v.10 and the Kistler 3-axis force dynamometer; aluminum upmilling at 600 RPM, no coolant	50
Figure 5.3 - Comparison of net force profiles between Smart Tool and Kistler 3-axis force dynamometer; aluminum upmilling at 3000, 3600, 4000 RPM, no coolant.....	51
Figure 5.4 - Dynamic chip load model [1].....	53
Figure 6.1 - Transfer function of an arbitrary linear system representing a measurement system	55
Figure 6.2 - Amplitude ratio of the arbitrary measurement system	55
Figure 6.3 - A measurement system whose dynamic parameters affect the accuracy of the measurement	56

Figure 6.4 - Generalized block diagram of the end-milling process.....	58
Figure 6.5 - Approach to interpreting the bending strain signal	59
Figure 6.6 - Block diagram of the autoregressive linear prediction model.....	61
Figure 6.7 - Inverse "spectral whitening" LPC model.....	62
Figure 6.8 – Model checking is performed by looking at the auto-correlation of the residuals	63
Figure 6.9 - 2 nd Order LPC model smears energy across the spectrum	65
Figure 6.10 - 4 th order LPC model describes resonance well	66
Figure 6.11 - 10 th order LPC model used to describe a system with two resonant modes.....	66
Figure 6.12 – De-trending the in-cut data for use with LPC.....	67
Figure 6.13 - 6th Order LPC works well for both free vibrations and de-trended force signals ...	67
Figure 6.14 – Smart Tool’s measured strain converted to force for an example cut where the force measurement has been corrupted by the system dynamics.....	68
Figure 6.15 - Welch power spectral estimate of the measured force signal	69
Figure 6.16 - LPC estimate of the power spectrum, P=6.....	69
Figure 6.17 - LPC estimate of the power spectrum, P=12.....	70
Figure 6.18 - LPC estimate of the power spectrum, P=32.....	70
Figure 6.19 - LPC estimate of the power spectrum, P=64.....	71
Figure 6.20 - LPC estimate of the power spectrum, P=128.....	71
Figure 6.21 - LPC estimate of the power spectrum, P=256.....	72
Figure 6.22 – LPC used to implement auto-regressive modeling of dynamic system resonance..	73
Figure 6.23 - Comparison of open-loop and closed-loop resonant frequencies	74
Figure 6.24 - Tracking variation in the fundamental resonant mode of tool vibration.....	75
Figure 6.25 - Time-lagged difference sequence represents the dynamic chip thickness	76
Figure 6.26 - Enhanced chatter frequency detection is achieved by differencing the measurement signal.....	77
Figure 6.27 - Comparison of differencing estimators.....	78

Figure 6.28 - Problem formulation for state estimation.....	80
Figure 6.29 - Discrete-time system and linear optimal filter. (a) Dynamic system and state estimator [11].....	82
Figure 6.30 - Discrete time system and linear-optimal filter. (b) Covariance estimator and gain computation [11].....	82
Figure 6.31 - Discrete Fourier Series representation of the static force profile.....	83
Figure 6.32 - Numerical integration of the static model spectrum is used to specify model order.....	84
Figure 6.33 - Frequency response of the Harmonic Kalman Filter	85
Figure 6.34 - Implementation of the Harmonic Kalman Filter.....	86
Figure 6.35 - The static model is used as a training signal to determine Wiener pole locations ...	91
Figure 6.36 - Z-Plane of the Wiener filter model	91
Figure 6.37 - Frequency response of the optimal Wiener solution.....	92
Figure 6.38 - Static signal estimate after applying the Wiener filter	93
Figure 6.39 - Output of the Wiener filter can be smoothed with a low-order MA filter	94
Figure 6.40 - Block diagram of the dynamic chip load filter.....	96
Figure 6.41 - Parameters for the dynamic chip load filter	96
Figure 6.42 - Dynamic chip load filter estimate of the applied cutting force	97
Figure 6.43 - Design parameters for the IIR notch filter	98
Figure 6.44 - Implementation of the IIR notch filter	99
Figure 6.45 - Zero-phase implementation of the IIR notch filter.....	99
Figure 6.46 - FIR notch filter implementation.....	100
Figure 7.1 – Suggested block diagram for real-time implementation of the Smart Tool	106
Figure D.1 – Block diagram of data reduction for net force comparison	138
Figure D.2 - Some bad Kistler data in the experiment.....	139
Figure D.3 - Blocking out sections of data to average.....	139
Figure D.4 - Identify peaks in the data set.....	141

Figure D.5 - Cycles aligned by peak value	141
Figure D.6 - Cross-covariance definition of the correlation coefficient used to align data	142
Figure D.7 - Aligned cycles are overlaid	142
Figure D. 8 - Aligned cycles are ensemble averaged to obtain the net profile	143
Figure D.9 - Net force profile comparison: 3000 rpm, half immersion.....	151
Figure D.10 – Net force profile comparison: 3000 rpm, three-quarter immersion.....	152
Figure D.11 - Net force profile comparison: 3600 rpm, half immersion.....	153
Figure D.12 - Net force profile comparison: 3600 rpm, three-quarter immersion	154
Figure D.13 - Net force profile comparison: 4000 rpm, half immersion.....	155
Figure D.14 - Net force profile comparison: 4000 rpm, three-quarter immersion	156
Figure E.1 – Radial strain converted to force	162
Figure E.2 – Power spectrum of the radial force measurement signal.....	163
Figure E.3 – LPC spectral estimate, 6th order LPC model.....	163
Figure E.4 – LPC spectral estimate, 12th order LPC model.....	164
Figure E.5 – LPC spectral estimate, 32nd order LPC model.....	164
Figure E.6 – LPC spectral estimate, 64th order LPC model.....	165
Figure E.7 – LPC spectral estimate, 128th order LPC model.....	165
Figure E.8 – LPC spectral estimate, 256th order LPC model.....	166
Figure E.9 - Overlay of force profiles aligned by the cross-correlation function	170
Figure E.10 – Average resonance as determined by LPC.....	172
Figure E.11 - Open loop and closed loop resonance as determined by LPC	175
Figure E.12 - Tracking resonance with LPC	176
Figure E.13 – Test for whiteness by using the autocorrelation.....	178
Figure E.14 - Static model used to train the Wiener filter	182
Figure E.15 - Zeros of the Wiener Filter.....	182
Figure E.16 - Frequency response of the Wiener filter.....	183

Figure E.17 - Implementation of the Weiner filter	184
Figure E.18 - Simulink model for the harmonic Kalman filter.....	188
Figure E.19 - Static force profile and DFS	189
Figure E.20 - Integration of the static model power spectrum.....	190
Figure E.21 -Implementation of the harmonic Kalman filter	191
Figure E. 22 - Bode plot of the harmonic Kalman filter	192
Figure E.23 – Example of Smart Tool data: Measured radial force, 3600 rpm.....	193
Figure E.24 – Parameters for the dynamic chip load filter	195
Figure E.25 – Implementation of the dynamic chip load filter	196
Figure E.26 – IIR notch filter z-plane and frequency response	198
Figure E.27 – Truncated impulse response used to build te FIR notch filter.....	199
Figure E.28 - Zeros of the FIR notch filter	199
Figure E. 29 - Frequency response of the FIR notch filter	200
Figure E.30 – IIR notch filter implementation.....	201
Figure E.31 - IIR notch filter, zero-phase implementation	202

LIST OF SYMBOLS

As Introduced by Chapter

Chapter 2

F_t	Tangential cutting force
F_r	Radial cutting force
F_{net}	Net cutting force
$\sigma_{bending}$	Bending stress at the Wheatstone bridge
M	Applied bending moment
y	Distance from neutral axis to point of stress analysis
I	Area moment of inertia
σ_{axial}	Axial stress at the Wheatstone bridge
F_{axial}	Axial force applied to the tool holder
A_c	Cross-sectional area of the tool holder
τ_{shear}	Shear stress at the Wheatstone bridge
V	Shear force
Q	First moment of area
t	Thickness of cross section at point of stress analysis
τ_{twist}	Torsional shear stress at Wheatstone bridge
T	Applied torque
r	Outside radius of the tool holder
J	Polar moment of inertia
ϵ_x	Strain in the x direction
ϵ_y	Strain in the y direction
ϵ_z	Strain in the z direction
σ_x	Stress in the x direction
σ_y	Stress in the y direction

σ_z	Stress in the z direction
ν	Poisson's ratio
E	Modulus of elasticity
γ_{xy}	Shear strain in the x-y plane
γ_{xz}	Shear strain in the x-z plane
γ_{yz}	Shear strain in the y-z plane
τ_{xy}	Shear stress in the x-y plane
τ_{xz}	Shear stress in the x-z plane
τ_{yz}	Shear stress in the y-z plane
G	Modulus of rigidity
δF	Incremental force (force resolution)
ΔE_o	Voltage output of the Wheatstone bridge
G_F	Gage factor
E_i	Wheatstone bridge excitation voltage
ε_1	Strain in bridge arm 1
ε_2	Strain in bridge arm 2
ε_3	Strain in bridge arm 3
ε_4	Strain in bridge arm 4
K_{inamp}	Gain of the differential instrumentation amplifier
δV	Incremental voltage (voltage resolution)
V_{sat}	Saturation voltage of the differential amplifier
φ	Circumferential misalignment angle
θ	Planar misalignment angle

Chapter 3

K_{sensor}	Lumped-parameter stiffness of the sensor
$K_{fixture}$	Lumped-parameter stiffness of the calibration fixture
u_{sensor}	Displacement of the tool tip
$u_{fixture}$	Displacement of the calibration fixture's free end

$K_{bending}$	Static bending sensitivity
K_{axial}	Static axial sensitivity
F	Static force measurement (either tangential or radial)
S	Static sensitivity (either tangential or radial)
ε	Measured strain (either tangential or radial)
ΔF	Total uncertainty in the static force measurement
$\Delta\theta$	Uncertainty in planar misalignment
Δl	Uncertainty in lever-arm distance
ΔS	Uncertainty in the static sensitivity
$\Delta\varepsilon$	Uncertainty in the measured strain

Chapter 5

F_x	X-component of force as measured by Kistler dynamometer
F_y	Y-component of force as measured by Kistler dynamometer
ρ_{xy}	Unbiased definition of the sample cross-correlation coefficient
R_{xy}	Unbiased definition of the sample cross-correlation
s_x	Standard deviation of arbitrary process X
s_y	Standard deviation of arbitrary process Y
τ	Correlation lag time
ω_d	Damped natural frequency
ζ	Damping ratio
ω_n	Natural frequency
k	Stiffness of an arbitrary system
m	Effective mass of an arbitrary system

Chapter 6

y	Observed continuous time process
x	True process
h	Impulse response of a linear system

$h_o(t)$	Static chip thickness
$h(t)$	Dynamic chip thickness
$F_c(t)$	Actual cutting force
$F_m(t)$	Measured cutting force
$\delta(t)$	Current tool deflection
$\delta_o(t)$	Tool deflection from previous tooth pass
$F(s)$	Transfer function relating actual cutting force to chip load
$G_1(s)$	Transfer function relating measured force to applied force
$G_2(s)$	Transfer function relating tool deflection to applied force
T	Time delay of previous tooth
P	Model order for linear filter
k	Indexing variable for filter coefficients
$H(z)$	Transfer function for linear filter
$A(z)$	Polynomial in z for Linear Prediction model
a_k	Auto-regressive model coefficients
$e(m)$	Discrete time error sequence
m	Discrete time data index
$x(m)$	Discrete sequence of true process
$\hat{x}(m)$	Discrete estimated sequence
$E[\]$	Expected value operator
r_{xx}	Autocorrelation vector
\mathbf{R}_{xx}	Autocorrelation matrix of absolute value of $r_{xx}(k-j)$
$J(x)$	Cost function of the state residual
x	Process to estimate
y	Process transformed by linear system
v	Additive white Gaussian noise
z	Observation sequence
\hat{x}	Estimated process
H	Observation matrix for an arbitrary state model

w_k	Weiner filter FIR-model coefficients
SSE	Sum of squared error
\hat{r}_{yy}	Sample autocorrelation (biased definition)
\hat{r}_{yx}	Sample cross-correlation (biased definition)
\hat{R}_{yy}	Sample auto-correlation matrix
Y	Matrix of observed process
x	Training signal vector
K_s	Static sensitivity
$\hat{y}(m - T)$	Estimated tool deflection (in strain)from previous tooth pass
$\hat{F}_C(m)$	Estimated cutting force

ABSTRACT

CALIBRATION AND CHARACTERIZATION OF A LOW-COST WIRELESS SENSOR FOR APPLICATIONS IN CNC END MILLING

By

Andrew James Harmon

University of New Hampshire, May 2012

Degree Advisor: Barry Fussell

Central to creating a smart machining system is the challenge of collecting detailed information about the milling process at the tool tip. This work discusses the design, static calibration, dynamic characterization, and implementation of a low-cost wireless sensor for end-milling. Our novel strain-based sensor, called the Smart Tool, is shown to perform well in a laboratory setting with accuracy and dynamic behavior comparable to that of the Kistler 3-axis force dynamometer. The Smart Tool is capable of measuring static loads with a total measurement uncertainty of less than 3 percent full scale, but has a natural frequency of approximately 630 Hz. For this reason, signal conditioning of the strain signal is required when vibrations are large.

Several techniques in signal processing are investigated to show that the sensor is useful for force estimation, chatter prediction, force model calibration, and dynamic parameter identification. The presented techniques include a discussion of the Kalman filter and Weiner filter for signal enhancement, Linear Predictive Coding for system identification, model-based filtering for force estimation, and sub-optimal linear filters for removing forced vibrations.

CHAPTER 1

INTRODUCTION

1.1 Introduction

The fundamental purpose of our work in the Design and Manufacturing Laboratory at UNH is to use engineering to make NC machining smarter. While the required technology to implement advanced controls in manufacturing has been available for decades, the machine shop industry has remained relatively static. Very little technology has been implemented in industry to automatically control part quality and production efficiency. Advances in manufacturing technology are of critical importance today because smart machining holds the promise of helping American companies stay competitive in a global economy.

1.1.1 The Importance of Making Observations

A smart machining system is characterized by its ability to operate and adapt to meet process objectives under uncertainty. Such a smart machining system would be capable of performing real-time Tool Condition Monitoring (TCM); it would make in-situ adjustments to feedrates and spindle speeds for process improvement subject to cost and objective functions, and it would be capable of monitoring dynamic stability for chatter prediction and control.

Central to performing these tasks, however, is the necessity to observe the milling process by making measurements and collecting data. Historically, the ability to record in-process data at the tool tip has been limited by the sensor location. Often, these sensors are located at significant physical distance from the cutting process [2]. Since most process objectives (such as tool deflection, chip thickness, limits for tool breakage, etc) can be directly related to cutting

force, determination of the cutting forces is of particular interest to developing an intelligent controller for CNC milling. Measuring instantaneous milling forces is a difficult problem, however, because the sensor must be non-invasive.

We have developed a novel, strain-based wireless sensor which we refer to as the “Smart Tool” (See Figure 1.1). Strain gages mounted on the tool holder body produce a signal proportional to cutting force when measured statically. Since milling forces are inherently non-static, the system dynamics must be carefully considered for accurate physical interpretation of the strain signal. In most cases, this signal can be used to estimate force; in situations where there is too much vibration to estimate force, the signal provides a means for investigating tool deflection, dynamic parameter identification, and potentially the onset of chatter.

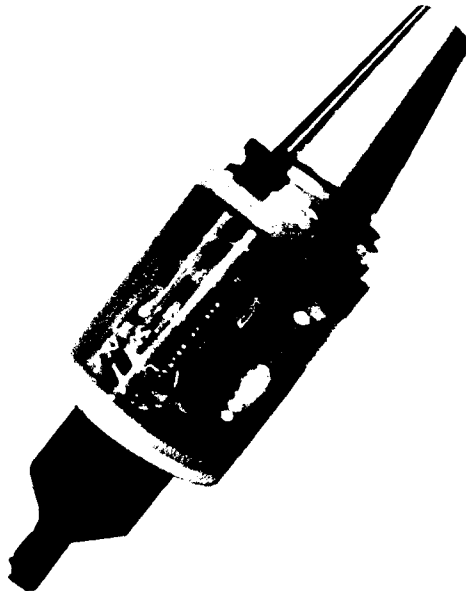


Figure 1.1 - Our Smart Tool sensor, version 10

1.2 Thesis Overview

Chapter 2 discusses the 10th generation design of our low-cost, wireless “Smart Tool”. Design criteria are developed, and specifications for the selected design are presented in detail. Chapter 2 addresses strain gage selection, placement, and the sensor’s mechanical theory of

operation. The analytical formulae for bending sensitivity and a comprehensive cross-sensitivity analysis are also presented. Chapter 2 concludes with a discussion of the data transmission board and conditioning electronics.

Chapter 3 presents results of the static calibration used to determine the sensor's static sensitivity and cross-sensitivities. These empirical values are necessary to convert the digital strain data to force. The cross-sensitivities are used to develop static confidence intervals for force measurements. This establishes a baseline for evaluating total measurement uncertainty.

Chapter 4 presents a dynamic characterization of the end-milling system. Open-loop poles of the static transfer function are determined experimentally, and variation in the damping ratio and natural frequency are investigated as a function of spindle speed. The problem of dynamic parameter identification is presented as a way of dealing with stochastic variation in system parameters.

Chapter 5 presents an experimental validation of our sensor by comparing Smart Tool v.10 to a Kistler 3-axis force dynamometer for a variety of cutting conditions. We show that both sensors accurately measure force when vibrations are small. The dynamic effects of the sensors' output signals are investigated when vibrations are significant. This chapter leaves us with the critical question, "How do we interpret the output of our sensor when the measurement is corrupted by system dynamics?" Answers to this question are provided in Chapter 6.

Chapter 6 is a survey of applications in signal processing that can be useful in interpreting the output signal of the Smart Tool. We present a simple algorithm for enhanced chatter frequency detection, and various filters are developed to remove unwanted system dynamics. Linear Predictive Coding (LPC) is shown to be useful for dynamic parameter estimation.

Chapter 7 provides conclusions for this work by thinking about how the sensor could be implemented for real-time control. Suggested direction for future work is also presented.

CHAPTER 2

WIRELESS SENSOR DESIGN

2.1 Introduction

The Smart Tool Project is a research initiative at the University of New Hampshire's Design and Manufacturing Laboratory focused on developing low-cost, wireless sensors for applications in Smart Machining [2, 6, 14]. Previous efforts in this research initiative have involved the design of sensors for high-bandwidth torsion data, and sensors for chatter detection [2, 6, 14]. Smart Tool v.10 is the result of a major redesign effort to eliminate cross-sensitivities to unwanted components of strain. This chapter outlines the design requirements and design specifications of Smart Tool v.10.

2.2 Project Statement

The objective of the Smart Tool Project v.10 is to design, build, and analyze a wireless strain sensor for CNC end milling that is:

- Robust
- Minimally invasive to the machining environment
- Sufficiently sensitive and accurate
- Stable with respect to time, temperature, and light
- Less expensive than industry alternatives

The successful design must accurately resolve its measurement from a combined loading scenario while remaining insensitive to unwanted components of strain.

2.3 Sensor Design Criteria

The qualitative design objectives delineated in the project statement are meant to ensure that the device is practical in a machining environment and useful as a laboratory instrument. To be useful, the sensor must lend itself to material characterization, force model calibration, chatter detection, and development of a real-time quality controller. A specific list of design criteria is presented below in Table 2.1:

Table 2.1 – Sensor Design Criteria

Category	Attribute	Specification
Physical Requirements	Sensor Capability	Measure instantaneous force, or strain signal adequate for state estimation
Functional Performance	Size Limitations	Overall length < 25.4 cm (10 in) Max diameter < than 15.24 cm (6 in)
	Resolution	Minimum resolution 4.5 N (1 lbf)
	Span	1330 N (300 lbf) at tool radius
	Data Collection	Use a wireless protocol
	DC Stability	DC drift less than 3 percent full scale with respect to time, temperature, and humidity
	Cross-Sensitivity	Bending crosstalk < 1% full scale Torsional crosstalk < 1% full scale
	Sampling Rate	3 kHz minimum sampling rate
	Precision and Repeatability	Identical loads must correspond to 99% repeatability
	Linearity	Linear over calibrated range
	Hysteresis	Less than 1 percent full scale
Total Error	Less than 5 percent full scale	
Operating Conditions	Spindle Speed	0 to 7500 RPM
	Process Factors	Sensor not affected by cutting fluid, temperature, and metal chips
Human Factors	User Controls	Intuitive and labeled
Aesthetic	Strain Display	8 LED linear display
Economic	Cost	Less than \$2,000
Safety	Structural Integrity	Safe to operate at all spindle speeds

2.4 Overview of Smart Tool v.10

A successful sensor design was chosen from several candidate designs. A summary of this design is presented below. The protocol for construction of Smart Tool v.10 can be found in Appendix B.

2.4.1 Macro-view Theory of Operation

Consider a combined load (bending and torsion) applied to the sensor as a result of an arbitrary milling operation: The net force acting on the end mill creates a stress field of normal and shear stresses through the tool holder body. This stress distribution leads to a strain distribution based on Hooke's Law and resulting tool deflection. Strain gages are carefully placed on the tool holder body to resolve the bending strains in the tangential and radial directions.

The voltage output from the strain gage Wheatstone bridge is proportional to strain, and this voltage is conditioned by an on-board instrumentation amplifier. The analog signal is next converted into a 16-bit digital signal and transmitted at 10.24 kHz via Bluetooth to a host PC over a serial connection. Currently, the wireless communication system is only capable of sampling and transmitting one channel of data. This means that while the sensor is outfitted with both tangential and radial strain gages, only one signal can be measured at any given time. A two-channel serial communication board should be developed as future work.

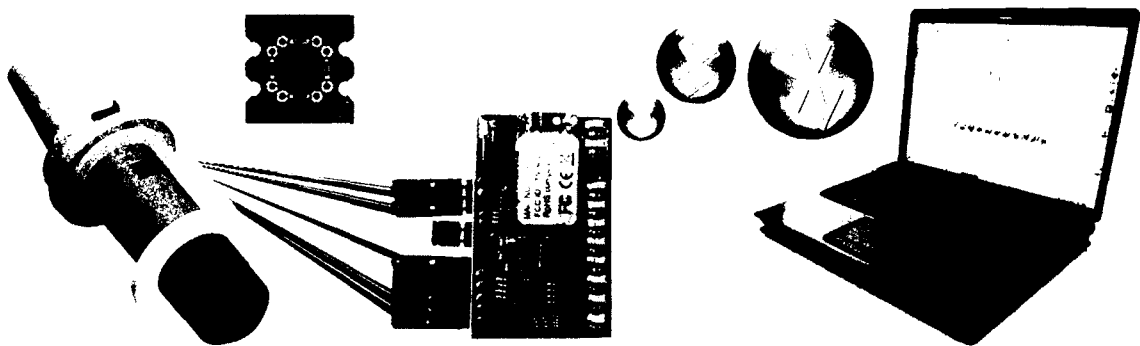


Figure 2.1 - Overview of the data transmission sequence

2.5 Strain Gage Selection

Semiconductor strain gages were chosen in this application because of their large gage factor, high bandwidth, low power consumption, and small physical footprint [5]. Careful strain gage placement allows the sensor to isolate bending strains associated with the tangential force from radial forces and torsion. Similarly, a second gage can isolate the strain effects from radial forces. The semiconductor strain gages are organized in a Wheatstone bridge and populated on a printed circuit board. To measure bending strains while mechanically avoiding other unwanted components of strain, the semiconductor strain gages are mounted orthogonally on the tool holder body near the tapered collet. Our semiconductor strain gages are highly sensitive with a nominal resistance of 350 ohms and a gage factor of approximately 140. Such a high sensitivity allows the sensor to accurately resolve strains on the order of 10^{-7} . Furthermore, the semiconductor gages in each bridge are thermally matched to have the same coefficient of thermal expansion. By matching the coefficients of thermal expansion, and by placing all arms of the bridge in close proximity, the sensor is designed to minimize any systematic bias created as a result of thermal gradients across the tool holder body.

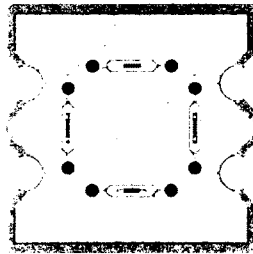


Figure 2.2 - Schematic of a semiconductor Wheatstone bridge purchased from Suprock Technologies

2.6 Orthogonal Decomposition of Strain

The sensor's theory of operation requires an orthogonal decomposition of tool deflection (i.e. strain) to reconstruct the net cutting force. Consider a single-tooth flat end mill removing material at a constant axial depth. The net force acting on the sensor is idealized as a planar force

because axial forces are small for inserts with a small helix angle. The net force acting on the engaged portion of the cutting tooth is resolved into orthogonal components by strain gages.

Because the cutting insert on the end mill has a helix angle of 14 degrees, there is no single position to mount the strain gages that will work for all axial depths of cut. The Wheatstone bridge is aligned such that the bridge center is nominally oriented with the cutting tooth for an axial depth of 3.18 mm (1/8 in). This means that the radial-bridge axis is centered above the point on the cutting insert 1.59 mm (1/16 in) from the insert tip. The tangential bridge is mounted orthogonal to the radial bridge. For axial depths of cut other than 3.18 mm, a slight trigonometric adjustment is required to account for the vector change in the net force due to the insert's helix angle. Figure 2.3 shows an idealized cross section of the tool holder body and illustrates gage layout with respect to the cutting tooth:

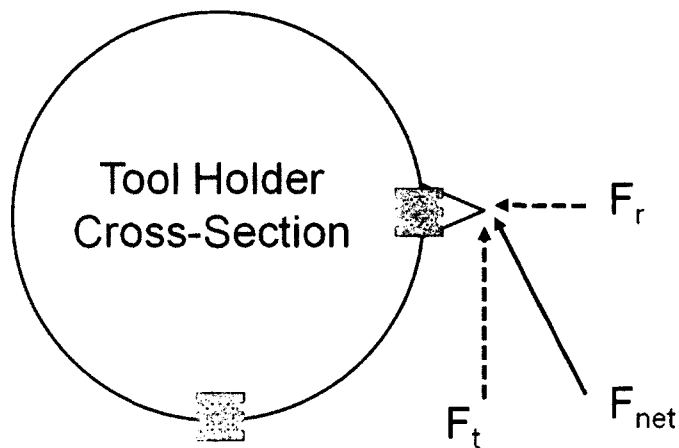


Figure 2.3 - Positioning of the Wheatstone bridges to isolate bending strains

Orthogonal positioning of the radial and tangential bridges allows the sensor to measure these components of strain independently. This occurs because the neutral axis for bending passes through the orthogonal gage location. With this configuration, the perpendicular components of bending strain are *mechanically decoupled*.

To understand the mechanics of how the sensor resolves bending strain from a combined loading scenario, consider the tool holder body modeled as a cantilever beam as shown in Figure 2.4. The outer diameter of the tool holder body is 31.8 mm (1.25 in), and it accepts a 19 mm (0.75 in) insert holder. The lever arm distance from the tool tip to the gages is approximately 131 mm (5.1 in).

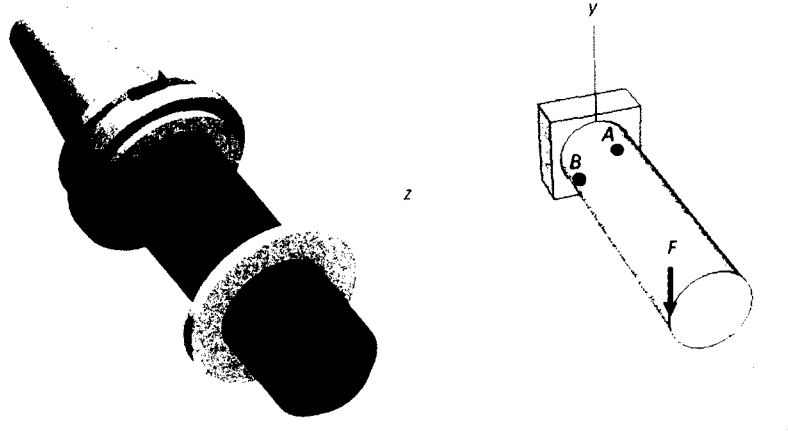


Figure 2.4 - Beam model illustrating orthogonal decomposition of bending strain

Neglecting the size of the strain gages and treating the entire Wheatstone bridge as a single point, we next consider an arbitrary force applied at the tool tip. Loading the structure in this way creates a stress distribution that can be decomposed into four stress components: Bending normal stress, axial normal stress, transverse shear stress, and torsional shear stress. These stress components are given by the familiar equations:

$$\sigma_{bending} = \frac{M \cdot y}{I} \quad (2.1)$$

$$\sigma_{axial} = \frac{F_{axial}}{A_c} \quad (2.2)$$

$$\tau_{shear} = \frac{V \cdot Q}{I \cdot t} \quad (2.3)$$

$$\tau_{twist} = \frac{T \cdot r}{J} \quad (2.4)$$

Here, M is the applied bending moment, y is the distance from the neutral axis, I is the area moment of inertia, F_{axial} is the axial force, A_c is the area of the tool holder cross section, V is the shear force, Q is the first moment of area, t is the thickness of the cross section, T is the torsional moment about the x-axis, r is the radius from the area centroid to the point of applied load, and J is the polar area moment of inertia.

Through Hooke's Law, these stress components can be directly transformed into strain:

$$\varepsilon_x = \frac{1}{E} [\sigma_x - \nu(\sigma_y + \sigma_z)] \quad (2.5)$$

$$\varepsilon_y = \frac{1}{E} [\sigma_y - \nu(\sigma_x + \sigma_z)] \quad (2.6)$$

$$\varepsilon_z = \frac{1}{E} [\sigma_z - \nu(\sigma_x + \sigma_y)] \quad (2.7)$$

$$\gamma_{xy} = \frac{\tau_{xy}}{G} \quad (2.8)$$

$$\gamma_{xz} = \frac{\tau_{xz}}{G} \quad (2.9)$$

$$\gamma_{yz} = \frac{\tau_{yz}}{G} \quad (2.10)$$

Young's Modulus and Poisson's Ratio (E, ν) are those for plain carbon steel.

Furthermore, recall that only normal stresses can cause a change in output of the Wheatstone bridge. Normal strains are capable of creating dimensional changes in the strain gage grid by the Poisson effect. Changes in gage length and cross section create a proportional change in the gage's electrical resistance. Shear strains, however, merely rotate the grid and do not cause the elongation or contraction necessary to vary the resistance [7].

The instantaneous value of strain is an intermediate parameter in force measurement, and for this reason, we do not elaborate on the stress/ strain transformation. While the Smart Tool's output is proportional to bending strain, what we are ultimately concerned with is the relationship

between cutting force and output bits from the A/D converter. This *static sensitivity* is determined through experimental calibration. For this reason, we do not need to know the strain magnitude directly.

In summary, orthogonal placement of the Wheatstone bridges causes the neutral axis of one bridge to pass through the gage location of the second bridge. It is this alignment of the gages with orthogonal neutral axes that mechanically decouples the components of bending strain. Of course, strain gage size effects, alignment imprecision, and transverse sensitivity can all contribute to cross-sensitivity between the radial and tangential signals. These effects are quantified through static calibration of the sensor (see Chapter 3).

2.7 Bending Sensitivity Analysis

A critical part of the sensor design is analytically determining the resolution and span of the instrument to ensure that the design is adequate. A derivation of the bending sensitivity calculation proceeds as follows:

Equation 2.5 relates strain to stress by Hooke's Law. If we consider the tool loaded in pure bending, Equation 2.5 reduces to the one-dimensional form of Hooke's Law:

$$\varepsilon_x = \frac{\sigma_x}{E} \quad (2.11)$$

Substituting the expression for bending stress in a beam, $\sigma_x = "My/I"$, we obtain:

$$\varepsilon_x = \frac{\delta F \cdot l \cdot r}{E \cdot I} \quad (2.12)$$

Equation 2.12 relates an arbitrary load at the end of the tool, δF , to the bending strain at the gage location. Here, E is the modulus of elasticity and I is the area moment of inertia; l is the distance from the applied load to the gages, and r is the outside diameter of the tool holder body.

The voltage output of the Wheatstone bridge due to the presence of bending strain is given by:

$$\Delta E_o = \frac{G_F \cdot E_i}{4} (\varepsilon_1 - \varepsilon_2 + \varepsilon_3 - \varepsilon_4) \cdot K_{inamp} \quad (2.13)$$

Where G_F is the gage factor of the semiconductor strain gages, E_i is the bridge excitation voltage, and K_{inamp} is the differential amplifier gain. Because two of the gages are dummy gages and see no change in strain since they are perpendicular to the bending load (and neglecting transverse sensitivity effects), the output of the Wheatstone bridge is only dependent on the strain in the active arms of the bridge (1 and 3). Equation 2.13 reduces to:

$$\Delta E_o = \frac{G_F \cdot E_i}{2} \cdot \varepsilon_x \cdot K_{inamp} \quad (2.14)$$

And substituting our expression for strain at the gage location, we obtain the relationship between an arbitrary load applied to the sensor and the corresponding output of the Wheatstone bridge:

$$\Delta E_o = \frac{G_F \cdot E_i}{2} \left(\frac{\delta F \cdot l \cdot r}{E \cdot I} \right) \cdot K_{inamp} \quad (2.15)$$

The smallest change in voltage that can be measured by the sensor is determined by the resolution of the A/D converter. With a 16 bit A/D converter, the smallest voltage that can be measured is given by:

$$\delta V = \frac{\text{span}}{\text{bits}} = \frac{3.5 \text{ V}}{2^{16} \text{ bits}} \quad (2.16)$$

By equating (2.15) and (2.16), we obtain:

$$\delta V = \Delta E_o = \frac{G_F \cdot E_i}{2} \left(\frac{\delta F \cdot l \cdot r}{E \cdot I} \right) \cdot K_{inamp}$$

We then solve the above equation for force:

$$\delta F = \frac{2 \cdot E \cdot I \cdot \delta V}{G_F \cdot E_i \cdot l \cdot r \cdot K_{inamp}} \quad (2.17)$$

Finally, we can change the δV from the smallest output of the sensor to the saturation limit of the sensor to determine the span of the instrument:

$$F_{max} = \frac{2 \cdot E \cdot I \cdot V_{sat}}{G_F \cdot E_i \cdot l \cdot r \cdot K_{inamp}} \quad (2.18)$$

Equations 2.17 and 2.18 express the relationship between various design parameters and the resolution and span of the sensor. This derivation for bending sensitivity neglects the stress

effects that arise from a combined loading scenario, and it ignores any transverse sensitivity of the dummy gages. Ultimately, these effects will be accounted for in the static calibration. Here, we need a reasonable order-of-magnitude calculation to ensure that we meet the design criteria for resolution and span specified in Table 2.1. The relevant design specifications for computation of δF and F_{max} are given in Table 2.2.

Table 2.2 - Table of design specifications for calculation of bending sensitivity

Parameter	Symbol	Specification
Young's Modulus	E	200 GPa
Area moment of inertia	I	$1.852 \times 10^{-7} m^4$
Voltage resolution of ADC	δV	$\frac{3.5 V}{2^{16} bits}$
Saturation voltage of ADC	V_{sat}	3.3 V
Gain of the instrumentation amplifier	K_{inamp}	Variable 28-1300
Gage factor	G_F	140
Bridge excitation voltage	E_i	3.3 V
Moment arm (load to gage site)	l	0.131 m
Outside radius of tool holder	r	0.022 m

Using the design specifications of Table 2.2, the theoretical and experimental values for force resolution and span are plotted versus instrumentation amplifier gain (See Figure 2.5). The experimental curves were obtained by scaling the theoretical curves by a correction factor to match experimental observation. This correction factor was obtained by determining the bending load resulting in saturation for gains of $K_{inamp} = 52, 520, \text{ and } 1300$. We observe that with our design specifications, the resolution of the sensor far exceeds the design requirement of 4.5 N (1 lbf) for all values of gain, and that the span of the sensor can be varied from approximately 58 N

to 2.67 kN (13 lbf to 600 lbf). Furthermore, by designing the bit resolution to be an order of magnitude better than the design requirement, we allow sufficient overhead to accommodate sensor noise.

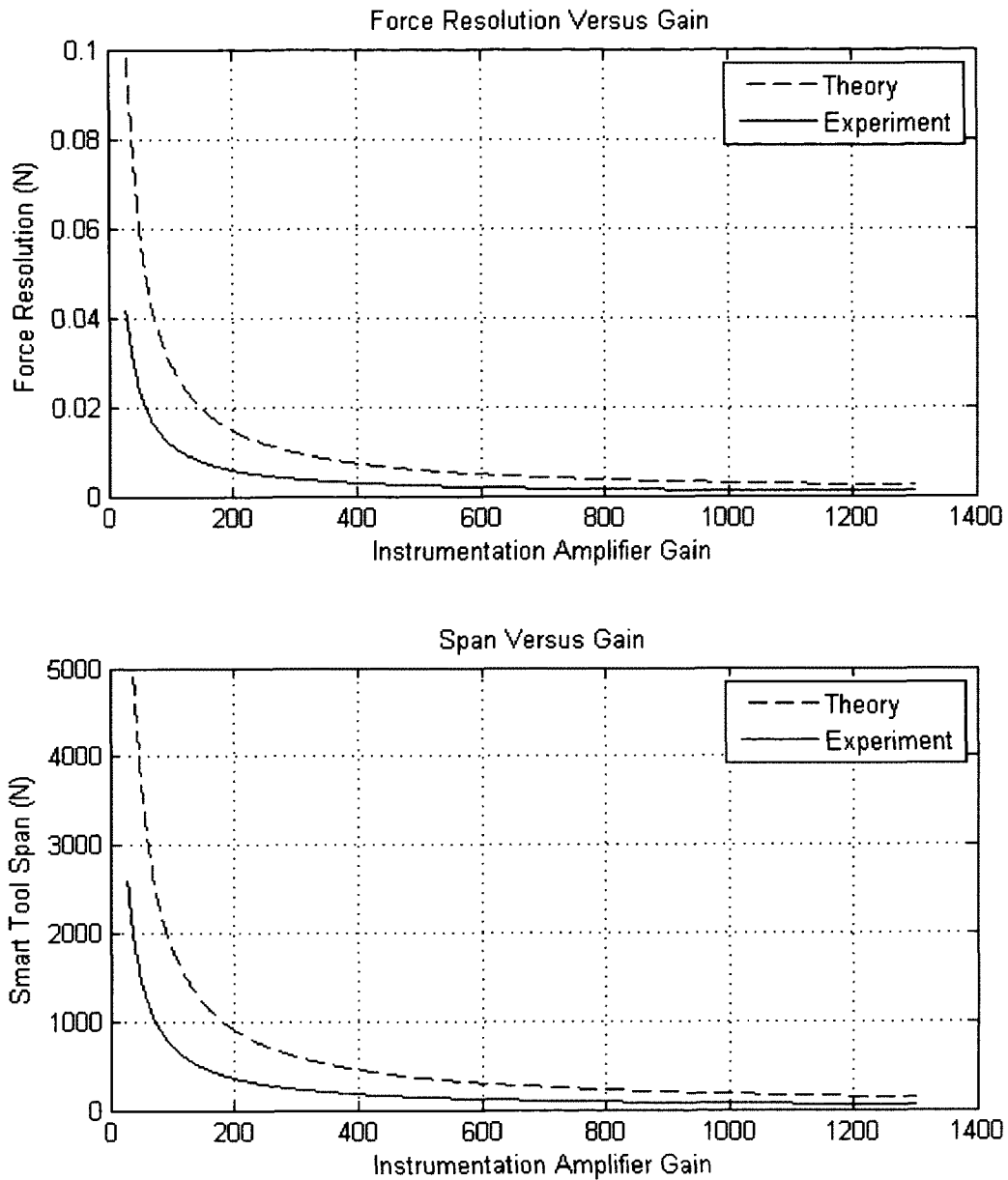


Figure 2.5 - Resolution and span versus instrumentation amplifier gain

This theoretical curve is proportional to $1/x$. The coefficient of proportionality is given by

$\frac{2 \cdot E \cdot I \cdot V_{sat}}{G_F \cdot E_t \cdot L \cdot r}$ as determined by Equation 2.17; the total uncertainty of this constant must account for

the difference between the theoretical and experimental curves seen above.

2.8 Cross-Sensitivity Analysis

Here, we investigate the effects of varying strain gage locations from their intended mounting positions; strain gage misalignment will result in indicated strain that does not reflect the actual strain of interest. Understanding the measurement error associated with strain gage misalignment allows us to determine the required accuracy during gage mounting to avoid cross-sensitivity errors. A derivation of the analysis is provided in Appendix C.

The analysis presented forthwith considers strain gages mounted on a hollow-cylindrical tool holder. This geometry is the ideal representation of the actual system and provides a first-order approximation of the measurement error produced by gage misalignment. The analysis shows that rotational misalignment around the circumference of the tool is more critical than misalignment in plane at a specific gage location. Furthermore, for circumferential misalignment (See Figure 2.1), measurement error is affected by the *ratio* of the tangential cutting force to the radial cutting force, and not by the magnitude of these forces. Measurement error produced by planar misalignment (i.e. torsional crosstalk) at a specific gage location is independent of the applied loading.

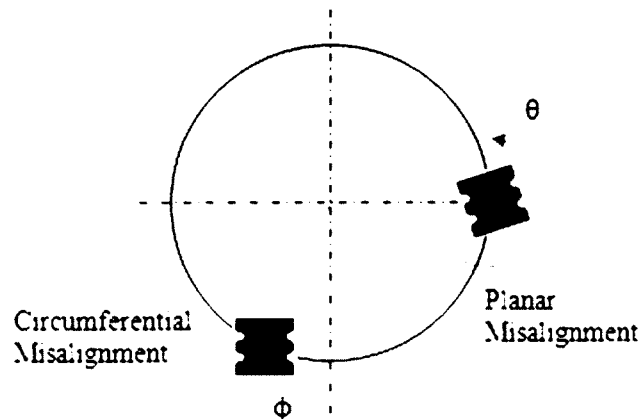


Figure 2.6 - Illustration of misalignment errors considered in this analysis

2.8.1 Assumptions

We make the following assumptions for the cross-sensitivity analysis:

1. The tool holder is a perfect hollow cylinder made of isotropic, plain carbon steel
2. The cutting insert is not modeled
 - a. Cutting forces are assumed to act at the end of the tool holder
 - b. Coupling effects (force transmission) between the insert holder and the tool holder are neglected
3. The gages are mounted on the curved cylindrical outer surface of the tool holder (i.e. not on a machined flat)
4. A Wheatstone bridge with two active arms is mounted at each gage location, nominally oriented to measure bending strains
5. Finite size effects of the strain gages are neglected (i.e. strain is a point measurement)
6. Transverse sensitivities and strain gage nonlinearities are not considered

2.8.2 Results and Discussion

Figure 2.7 shows the measurement error for the radial bridge versus circumferential misalignment for various ratios of tangential force to radial force. This measurement error is a result of bending crosstalk created by moving the bridge away from the neutral axis. Observe that the measurement error is proportional to the ratio of the cutting forces, and not to the magnitude of the forces themselves. Gage misalignment produces an indicated strain as a result of cross-sensitivity to the unwanted component of strain produced by the tangential force.

Figure 2.8 shows measurement error on the tangential bridge versus circumferential misalignment for various ratios of tangential force to radial force. Again, the measurement error is proportional to the ratio of these cutting forces, and not to the magnitude of the forces themselves. The measurement error increases as a function of misalignment, but decreases as a function of increasing ratios.

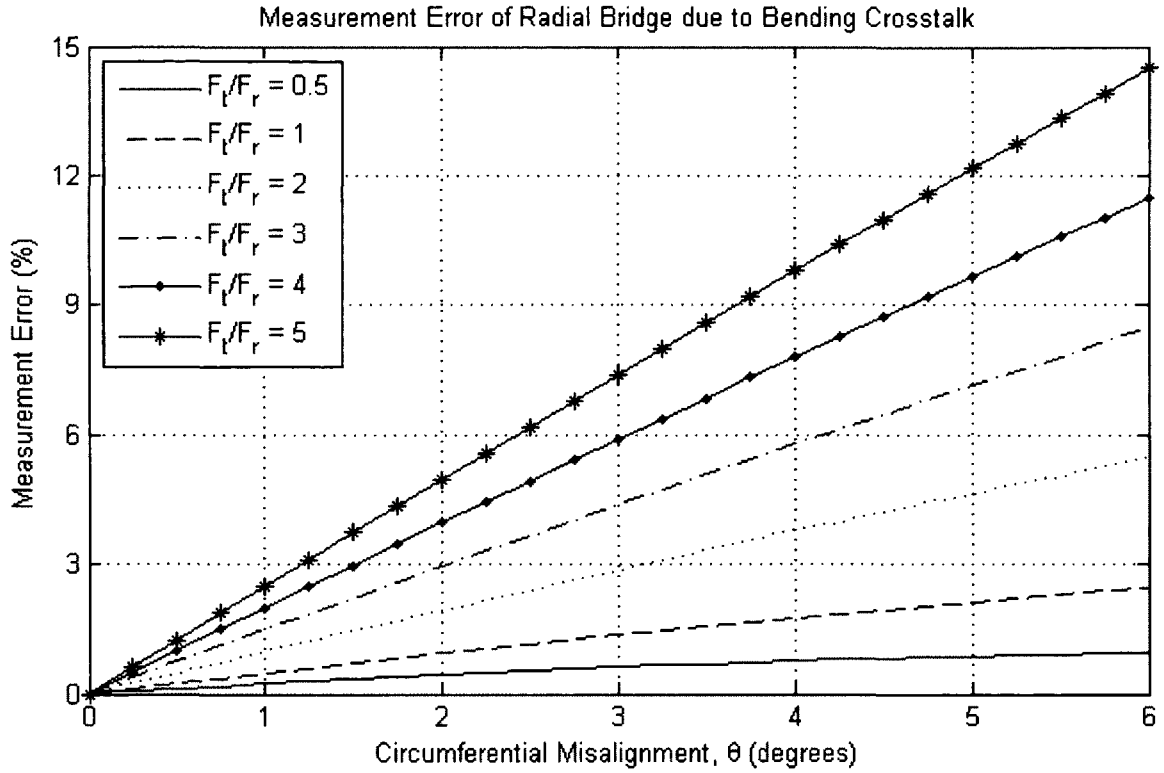


Figure 2.7 – Theoretical measurement error on the radial bridge due to bending crosstalk

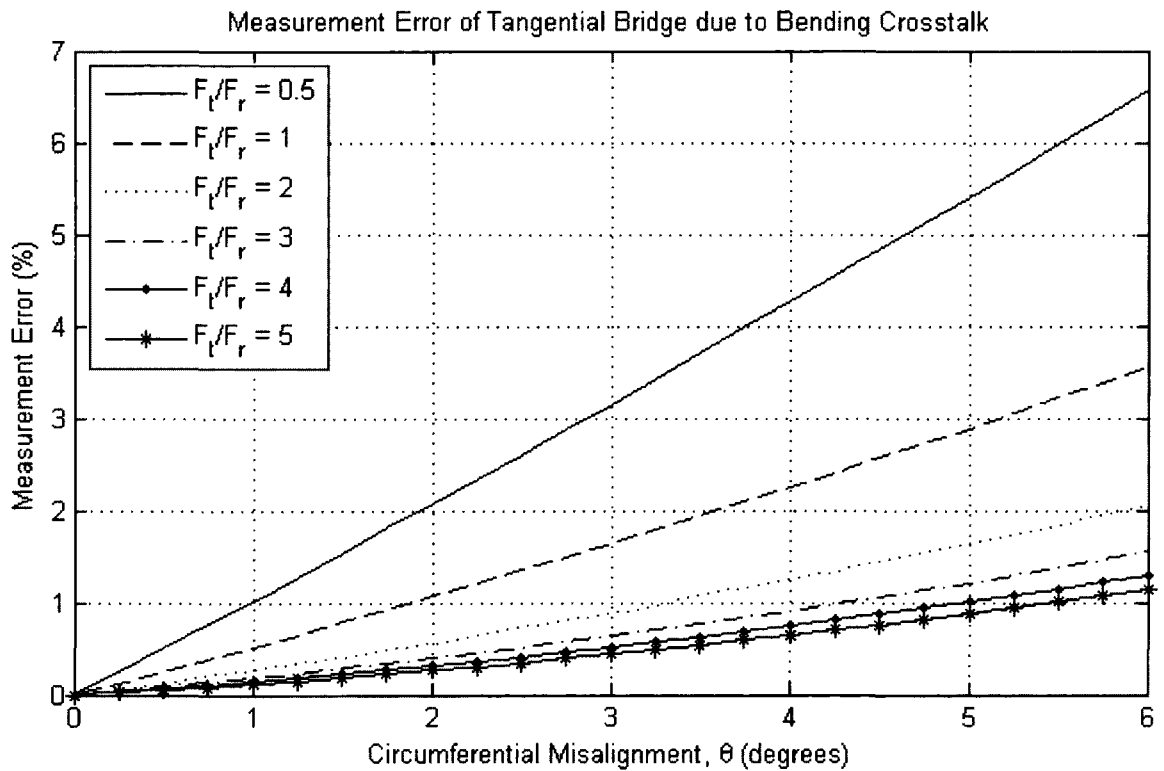


Figure 2.8 - Theoretical measurement error on the tangential bridge due to bending crosstalk

The third case of misalignment to consider is rotational misalignment of the Wheatstone bridge at the gage site. Figure 2.9 shows the effect of rotating a gage within the biaxial stress field at a specific gage location. Here, measurement error is observed to be independent of the loads applied on the tool holder, and is thus only affected by misalignment angle. Furthermore, because the sine of a small angle is quite small, it takes a substantially larger planar misalignment to produce a large measurement error.

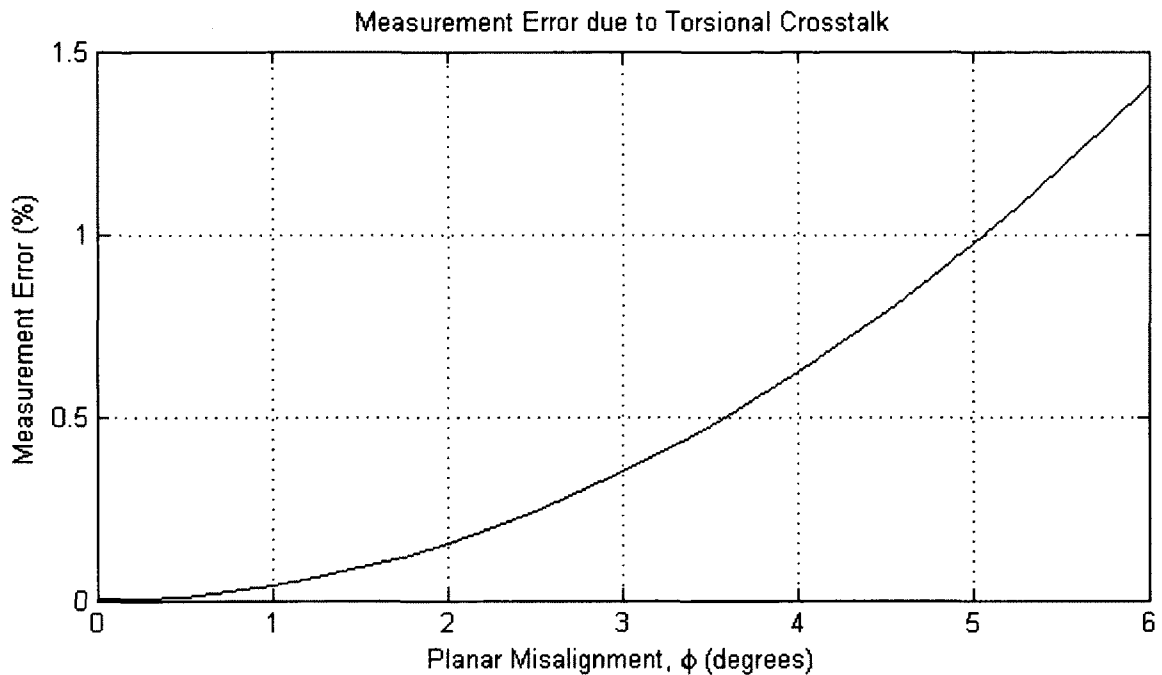


Figure 2.9 - Torsional crosstalk as a result of gage misalignment

Our analysis shows that rotational misalignment around the circumference of the tool holder is more critical than misalignment in plane at a specific gage location. Furthermore, for circumferential misalignment, measurement error is affected by the ratio of the tangential cutting force to the radial cutting force, and not by the magnitude of these forces. Error produced by planar misalignment at a specific gage location is independent of the loading applied to the tool holder. This analysis is a good first-order approximation of the expected error due to misalignment, however, it does not address issues related to transverse sensitivity, thermal gradients, or stress concentrations created by geometry of the tool holder.

2.9 Sampling and Data Transmission

Data transmission is accomplished by a small, low-power serial communication board. The current data transmission board was developed for a single-channel application [6] and is only capable of transmitting either radial or tangential strain via a manual switch. This switch is mounted in the lower end-cap of the shrouding and enables the user to choose which data set is transmitted to the computer.

The serial board is powered by a 3.71 V Lithium-polymer battery with an 850 mAh capacity. With a full charge, the sensor can continuously transmit data for approximately five hours before the low voltage dropout regulator turns off the device. The battery is then recharged by accessing a charging jack hidden beneath the retention bolt.

The analog voltage across the Wheatstone bridge is measured by an instrumentation amplifier, and sampled by a 16-bit analog to digital converter at 10.24 kHz. An LED array provides visual feedback about the sensor's ability to balance the bridges at startup, and after initialization, the height of the array serves as a visual indication of the measured strain. The data transmission board is shown below in Figure 2.10.

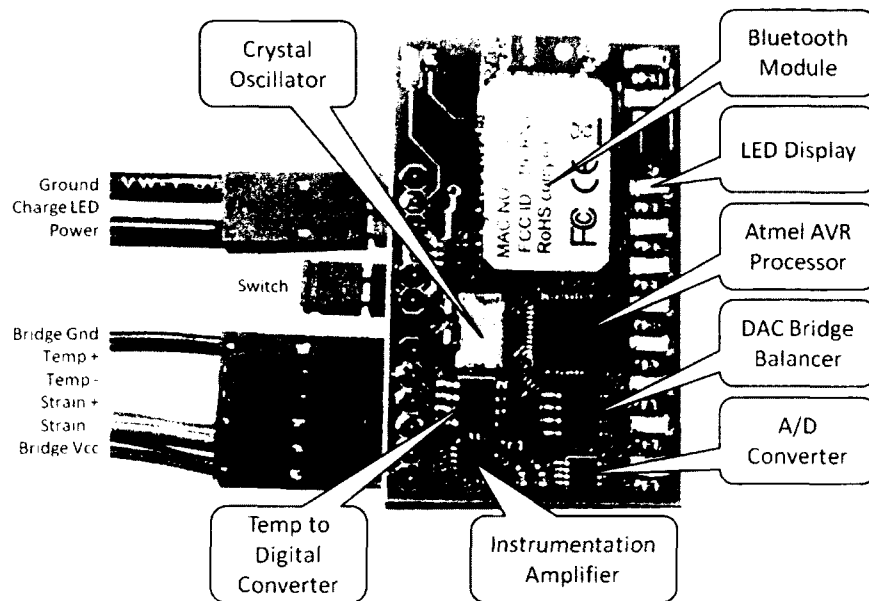


Figure 2.10 - Data transmission board designed by Jeffery Nichols [6, 14]

2.10 Summary

Design specifications for Smart Tool v.10 are presented in this chapter. A theoretical analysis of the bending sensitivity and span shows that the selected design specifications meet the required design criteria from Table 2.1. The noise-free force resolution is designed to be two orders of magnitude better than the design requirement of 4.5 N, thus allowing sufficient overhead for both noise on the received signal and the presence of minimal cross-sensitivity. The corresponding span is adjustable by using a digitally programmable instrumentation amplifier with variable gain.

A theoretical analysis of the indicated-strain measurement error is also presented. Indicated strain may not correspond to the true strain-state of interest if the gages are not precisely mounted during sensor construction. This analysis uses mechanics to predict the state of stress at the gage locations arising from the combined loading scenario for various ratios of tangential force to radial force. Stresses are converted to strains through the appropriate transformation equations, and the resulting strains are evaluated in the governing Wheatstone bridge equations. The outputs of the Wheatstone bridge for the misaligned scenarios are compared to the theoretical output for the perfectly-aligned case to predict measurement errors from cross-sensitivity. We observe that the strain gages need to be mounted within approximately 3 degrees of precision to keep theoretical measurement errors sufficiently small.

CHAPTER 3

STATIC CALIBRATION

3.1 Introduction

As stated in Chapter 2, the fundamental design objective for Smart Tool v.10 is to minimize bending and torsional cross-sensitivity errors to less than 1 percent full scale. While the analyses presented in Chapter 2 provide a theoretical framework for the required design specifications, experimental testing and calibration provide the true measure of design success. This chapter presents the experimental results of the static calibration performed to characterize static sensitivity and cross-sensitivity to both bending loads and torsion loads. The result of this calibration is a set of static confidence intervals that account for all possible sources of measurement error. These confidence intervals allow us to ensure that Smart Tool v.10 adequately meets the design requirement of achieving total measurement error less than 5 percent full scale for all loading scenarios. Protocols for the static calibration experiments can be found in Appendix A.

3.2 Calibration of Bending Sensitivity

Recall that semiconductor strain gages are oriented on the tool holder body to measure bending strains in the radial and tangential directions. The static bending sensitivities must be empirically determined to relate the sensor output signal to the applied load. This relationship is easily determined for static loads because the sensor output is directly proportional to the applied force. Using a servo-hydraulic Instron 55s machine, the sensor was loaded and unloaded 10 times to work out any initial hysteresis. Next, the bending sensitivity calibration was repeated 5 times to

determine the static sensitivity of the sensor in both the radial and tangential directions. The experimental setup is shown in Figure 3.1 and a diagram of the load transmission can be seen in Figure 3.2.



Figure 3.1 – Instron servo-hydraulic machine used to apply bending moments to the sensor

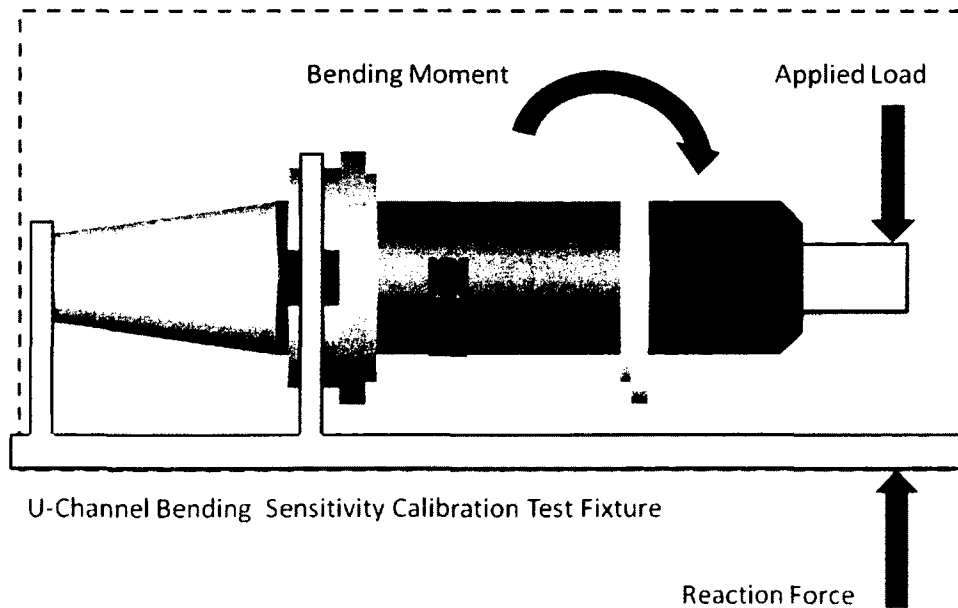


Figure 3.2 – Schematic of the load transmission through the sensor using the C-channel calibration fixture

The load transmission can be thought of as two springs in series, thus, the stiffness of the U-channel fixture does not affect the calibration results. As seen in Figure 3.3, the entire load is transmitted through both the sensor and the fixture.

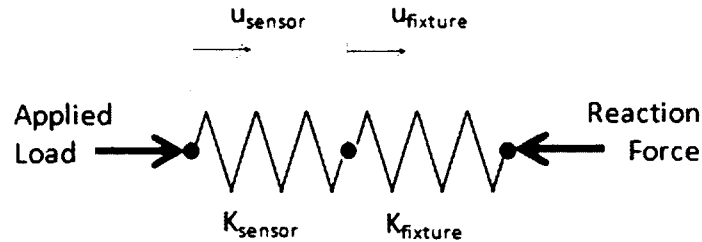


Figure 3.3 – Lumped parameter schematic of the Instron load train

Since the net displacement is affected by the stiffness of the fixture, we cannot calibrate strain. The static sensitivity is therefore determined by calibrating the output of the sensor (strain in bits) versus bending moment. This bending sensitivity calibration was repeated five times on each bridge with nearly identical results. The resulting calibration curves are shown in Figure 3.4 and Figure 3.5. Since the moment arm for calibration is different than the moment arm when the insert holder is in the tool, the x-axes for these graphs were calibrated by dividing the bending moment during testing by the moment arm during cutting (i.e. the distance from the cutting tooth to the center of the Wheatstone bridge).

We observe that the tangential and radial bridges have nearly identical sensitivities which differ by less than 0.6%. The y-intercept of approximately 32,000 bits is a result of the conditioning electronics that set the no-load output to half of the measurement range. This allows the sensor to measure both positive and negative bending strain. On startup, the output of the Wheatstone bridge is nominally biased to $2^{16} / 2$, which equals 32,768 bits. The ability of the sensor to zero itself at exactly half of the measurement range is limited by the resolution of the 12 bit DAC used to balance the Wheatstone bridge. Achieving a perfect bias also requires a state of exactly zero stress during initialization of the electronics at startup. We see that the sensor does an adequate job of nominally initializing the no-load output to half of the measurement range.

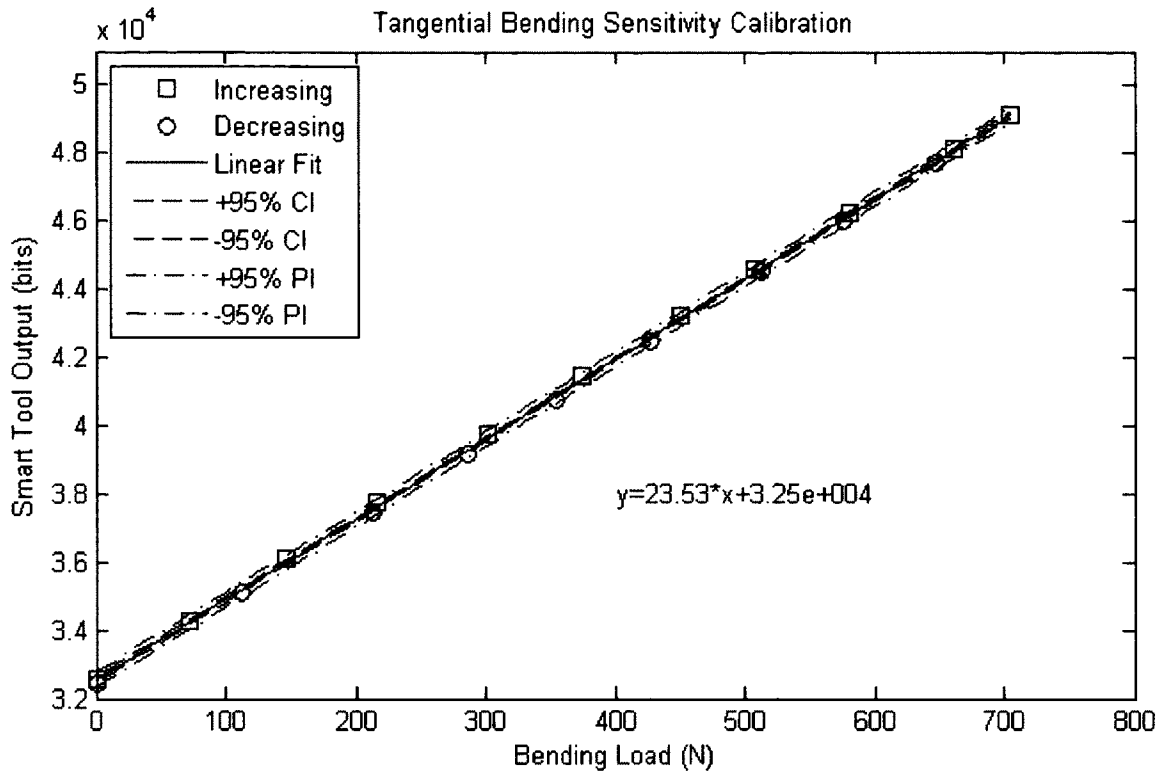


Figure 3.4 – Calibration curve for static bending sensitivity in the tangential direction

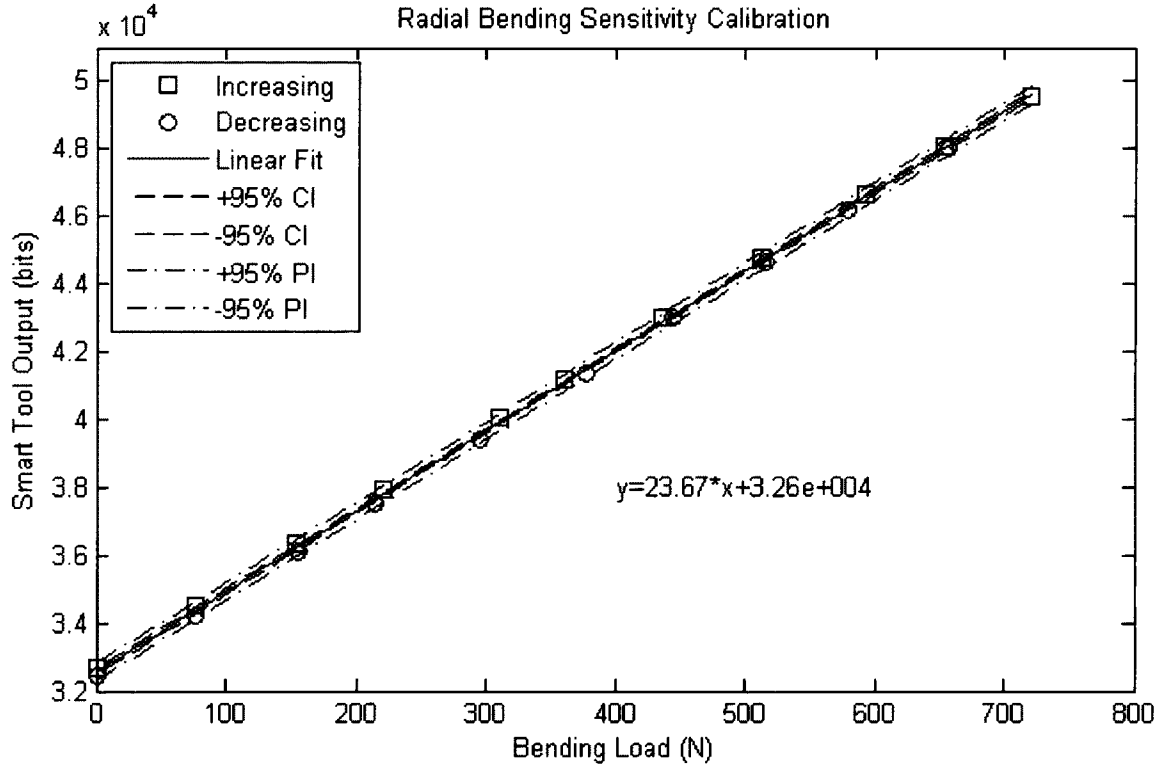


Figure 3.5 – Calibration curve for static bending sensitivity in the radial direction

From the bending sensitivity calibration, we can experimentally determine the span and the resolution of the sensor. Note that for all static calibration experiments, the gain of the instrumentation amplifier, K_{inamp} , was set to 52. The span of the sensor is given by:

$$Span = \frac{32,768 \text{ bits}}{23.5 \text{ bits}/N} = 1395 N \text{ (313 lbf)} \quad (3.1)$$

And the resolution can be determined as:

$$Resolution = \frac{1}{23.5 \text{ bits}/N} = 0.043 N/bit \text{ (} 9.56 \times 10^{-3} \text{ lbf/bit)} \quad (3.2)$$

While these empirical values are smaller than those predicted by the Sensitivity Analysis in Section 2.7, we find that using a gain of $K_{inamp} = 52$ adequately meets the design criteria presented in Table 2.1. Because the gain of the instrumentation amplifier is programmable, the sensor also has the flexibility to decrease span and increase resolution for lighter cuts.

3.3 Calibration of Bending Crosstalk

As described by the Cross-Sensitivity Analysis of Section 2.8, misalignment of the Wheatstone bridge with respect to the neutral axis results in measurement error due to bending crosstalk. This concept is illustrated by Figure 3.6:

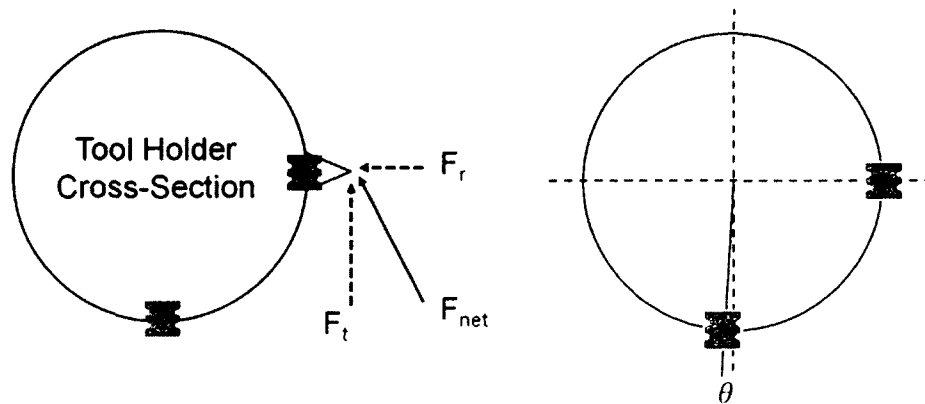


Figure 3.6 - Bending crosstalk results from circumferential misalignment of the Wheatstone bridge

Here, gage misalignment results in measurement error because the indicated strain now depends on both the tangential force and the radial force, and these signals become linear dependent. This type of measurement error is referred to as *bending crosstalk* and is calibrated statically to determine the coupling between the radial and tangential bending strain signals.

Calibration of bending crosstalk was accomplished by loading the sensor in pure bending on the transverse axis while measuring output from the Wheatstone bridge on the neutral axis. The experimental setup was identical to that for calibration of bending sensitivity, except now we measure the output of the bridge orthogonal to the loading. Experimentation was repeated five times using the Instron 55s servo-hydraulic machine. Full-scale calibration was achieved by applying bending moments equivalent to 300 lbf at the tool tip. A summary of the bending crosstalk is presented in Table 3.1:

Table 3.1 - Summary of bending crosstalk calibration

Measurement Bridge	Full Scale Load	Measured Crosstalk	Percent Full Scale
Radial	300 lbf at 5.17 in	166 bits	$\frac{166}{32768} \cdot 100 = 0.506 \%$
Tangential	300 lbf at 5.17 in	877 bits	$\frac{877}{32768} \cdot 100 = 2.68 \%$

As a point of interest, the cross-sensitivity analysis of Chapter 2 allows us to use this crosstalk data to estimate rotational misalignment of the strain gages with respect to the neutral axis. Using Figure 2.7, the radial bridge is presumably mounted with less than 1 degree circumferential misalignment from the neutral axis. The tangential bridge is mounted a bit further off-center with a misalignment of approximately 3 degrees as determined by Figure 2.8.

Also, while the tangential bridge is farther from the neutral axis than the radial bridge, measurement error on the tangential bridge is mitigated by the fact that the tangential force is often larger than the radial force. Therefore, the actual crosstalk (i.e. not full scale crosstalk) will be reduced.

3.4 Calibration of Torsional Crosstalk

While strain gages are insensitive to shear stresses, rotational misalignment of the Wheatstone bridge will cause the sensor to exhibit a cross-sensitivity to torsion. This occurs because rotation of the gage results in a strain transformation that causes the sensor to measure a component of the shear stress as a normal stress. This is shown pictorially in Figure 3.7.

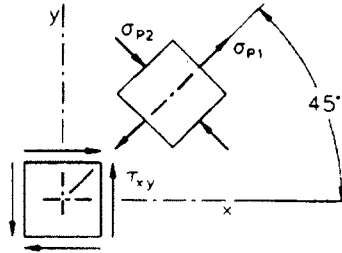


Figure 3.7 - Illustration of stress transformation resulting from rotation of the gage element

Normal strains cause dimensional changes in the grid of a strain gage, thus changing its electrical resistance. Pure shear strains merely rotate the grid, and do not cause the elongation or contraction necessary to vary the resistance [7]. Thus, if the strain gages are perfectly aligned, the sensor will be insensitive to torsional loads.

To calibrate torsional cross-sensitivity, the sensor was loaded and unloaded five times in pure torsion. To realize pure torsion, a moment was applied between two pillow block bearings. A flexible coupling was placed between the sensor and the static end of the load train to compensate for shaft misalignment. A picture of this calibration fixture is shown in Figure 3.8.



Figure 3.8 - Static calibration test fixture for torsional loading

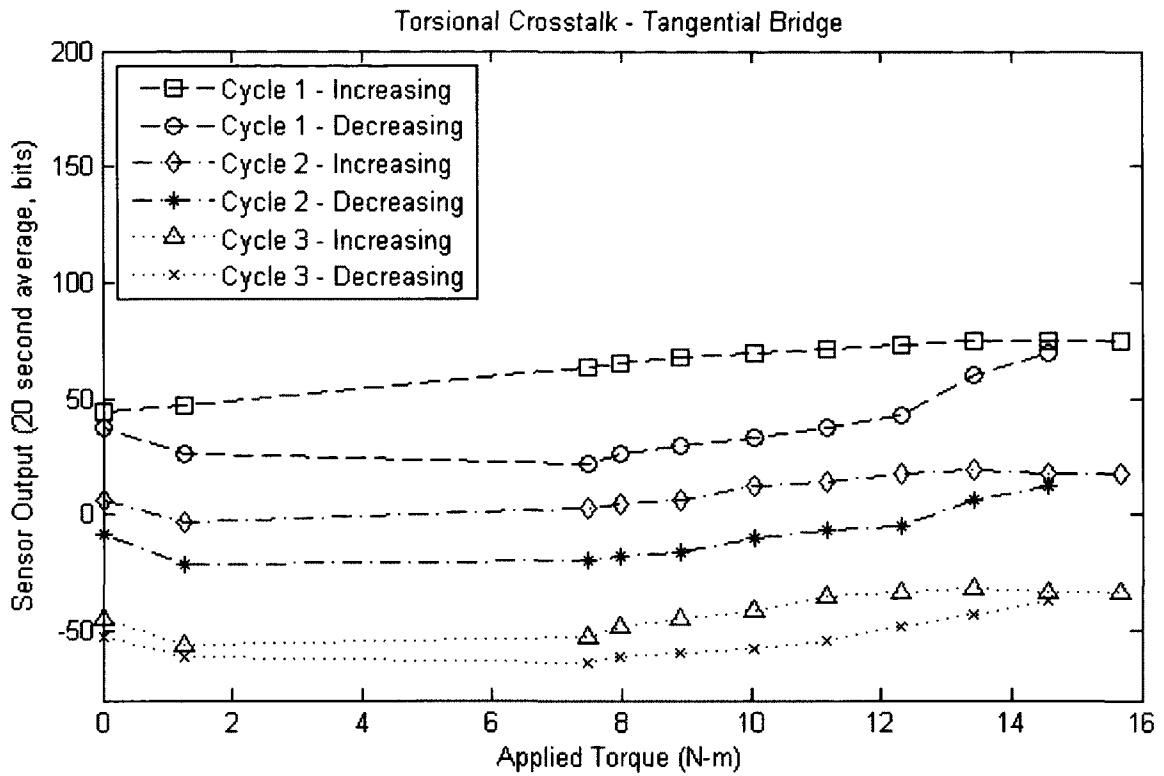


Figure 3.9 – Calibration of torsional crosstalk on the tangential bridge

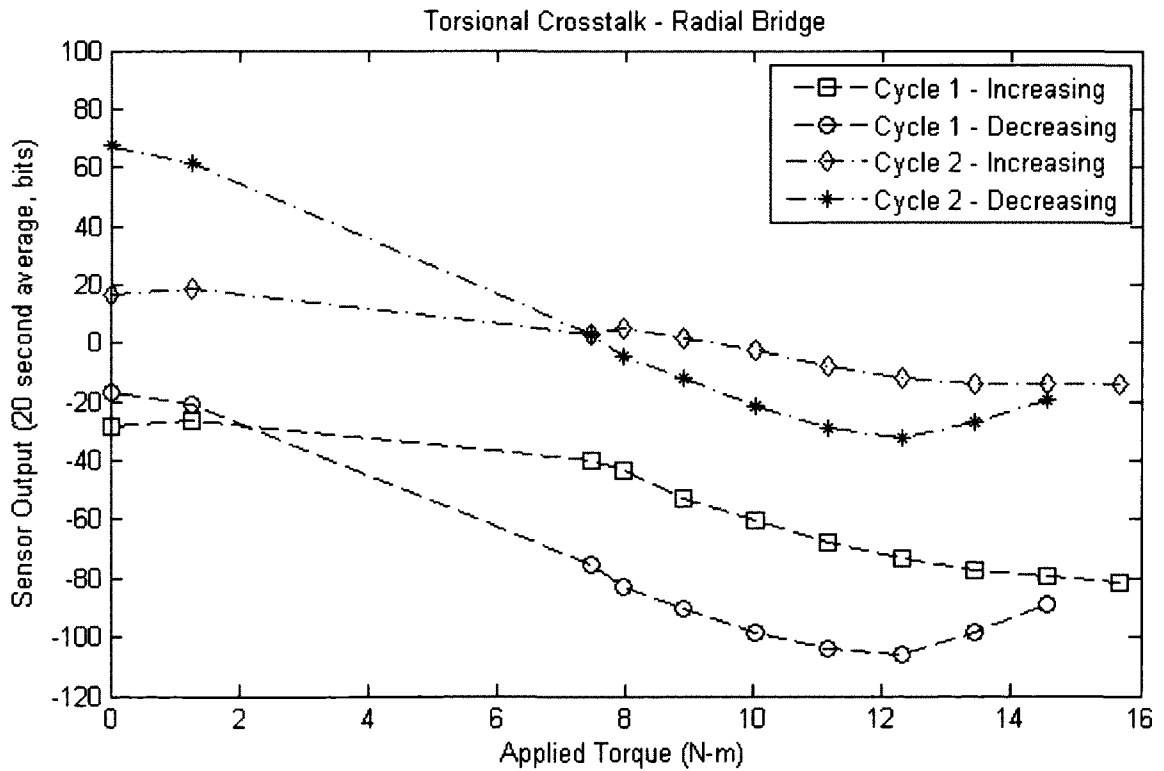


Figure 3.10 - Calibration of torsional crosstalk on the radial bridge

Figures 3.9 and 3.10 show the experimental results of the torsional cross-sensitivity calibration. It is observed that full scale torsional loading causes almost no change in the sensor output. By calibration of the static bending sensitivity, the sensor is observed to saturate at 1388 N (312 lbf): With a tool radius of 9.525 mm (0.375 in), the corresponding full-scale torsional loading is determined to be approximately 13.22 N-m. The Smart Tool was calibrated beyond this full range to approximately 16 N-m.

The vertical offset observed between loading cycles is an artifact of the Smart Tool's state of stress at startup; the force of gravity is large enough to change the strain at the gage locations and will bias the sensor output by a few bits. It is simple to compensate for this bias in post-processing. Also, for each observation, a 20 second moving average window was arbitrarily chosen to increase measurement confidence. A summary of the calibration is shown below in Table 3.2.

Table 3.2 - Summary of torsional crosstalk experimentation

Measurement Bridge	Full Scale Load	Measured Crosstalk	Percent Full Scale
Radial	312 lbf at 0.375 in	< 120 bits	$\frac{120}{32768} \cdot 100 = 0.37 \%$
Tangential	312 lbf at 0.375 in	< 50 bits	$\frac{50}{32768} \cdot 100 = 0.15 \%$

Static calibration of the torsional cross-sensitivity shows that the sensor exhibits significantly less than 1 percent torsional crosstalk full scale. Thus, we have satisfied the design requirement for torsional crosstalk presented in Table 2.1. We also observe that Figures 3.9 and 3.10 exhibit significantly more hysteresis than the results of the bending calibration. This is because these torsional loads were the first full-scale loads seen by the strain gages. Hysteresis is normal for initial loading cycles of strain gages [5] and is significantly reduced after a few full-scale loading cycles. There is no need to repeat this experiment, however, because even with this initial hysteresis, torsional cross-sensitivity is sufficiently small.

Again as a point of interest, Figure 2.9 allows us to use the cross-sensitivity analysis of Chapter 2 to infer that both bridges are mounted with better than 3 degrees of planar misalignment at their respective gage locations.

3.5 Calibration of Axial Sensitivity

Axial forces also create normal stresses through the tool holder body. Our strain gage configuration is sensitive to these normal stresses. In order to cancel axial stresses, all arms of the Wheatstone bridge must see the same change in resistance when loaded axially. This is not achieved with our sensor design because two of the bridge arms are perpendicular to this loading. Thus, the axial sensitivity was calibrated by suspending weights from the toolholder in the axial direction (See Figure 3.11). For a helix angle of 14 degrees, full scale axial loading is given by

$$F_{Axial,FS} = (1388 N) * \sin(14^\circ) = 336 N \text{ (75 lbf)} \quad (3.3)$$

Axial sensitivity was only calibrated over one-third of this range because of the weights that were readily available. Notwithstanding, this partial calibration remains sufficient to characterize axial sensitivity because the strain gages were already shown to be linear for full-scale bending strains.

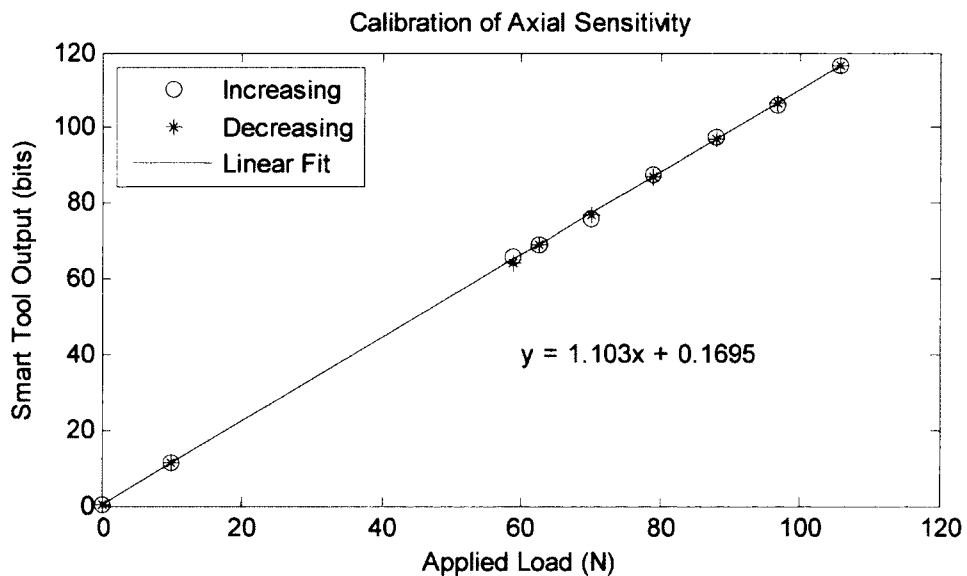


Figure 3.11 - Calibration curve for axial sensitivity of the Smart Tool v.10

The axial sensitivity of the sensor was determined to be 1.1 bits/N through static calibration. As shown by Equation 3.4, this means that the sensor is approximately 21 times more sensitive to bending strains than it is to axial strains.

$$\frac{K_{bending}}{K_{axial}} = \frac{23.53 \text{ bits}/N}{1.103 \text{ bits}/N} = 21.33 \tag{3.4}$$

Furthermore, even though the bending sensitivity is already 21 times larger than the axial sensitivity, the actual crosstalk from axial forces is further reduced because the axial force is always smaller than the in-plane forces. Theoretically the axial force is proportional to the net in-plane force by the sine of the helix angle. In practice, it is often even smaller. The X, Y, and Z forces are shown below for an upmilling operation at 600 RPM. The data was collected using a Kistler 3-axis bed dynamometer. This figure serves to further illustrate the insignificant nature of the axial force in milling.

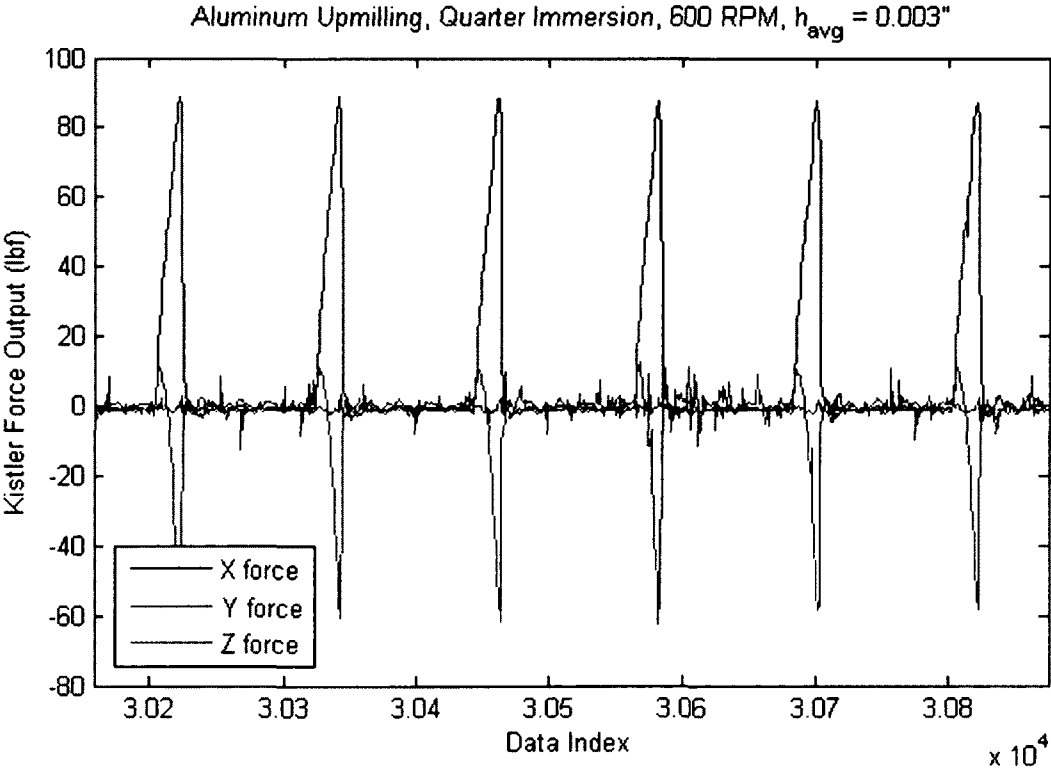


Figure 3.12 - Comparison of the axial force to the in-plane X and Y cutting forces

3.6 Characterization of Sensor Noise

Noise on the strain signal reduces the effective resolution of the sensor. Possible sources of noise are discussed in [14] by Jeff Nichols, designer of the data transmission board used on Smart Tool v.10. To characterize the noise, Figure 3.13 shows a time plot of the noisy signal and both the probability density function (PDF) and the cumulative probability density function (CPDF) for this signal.

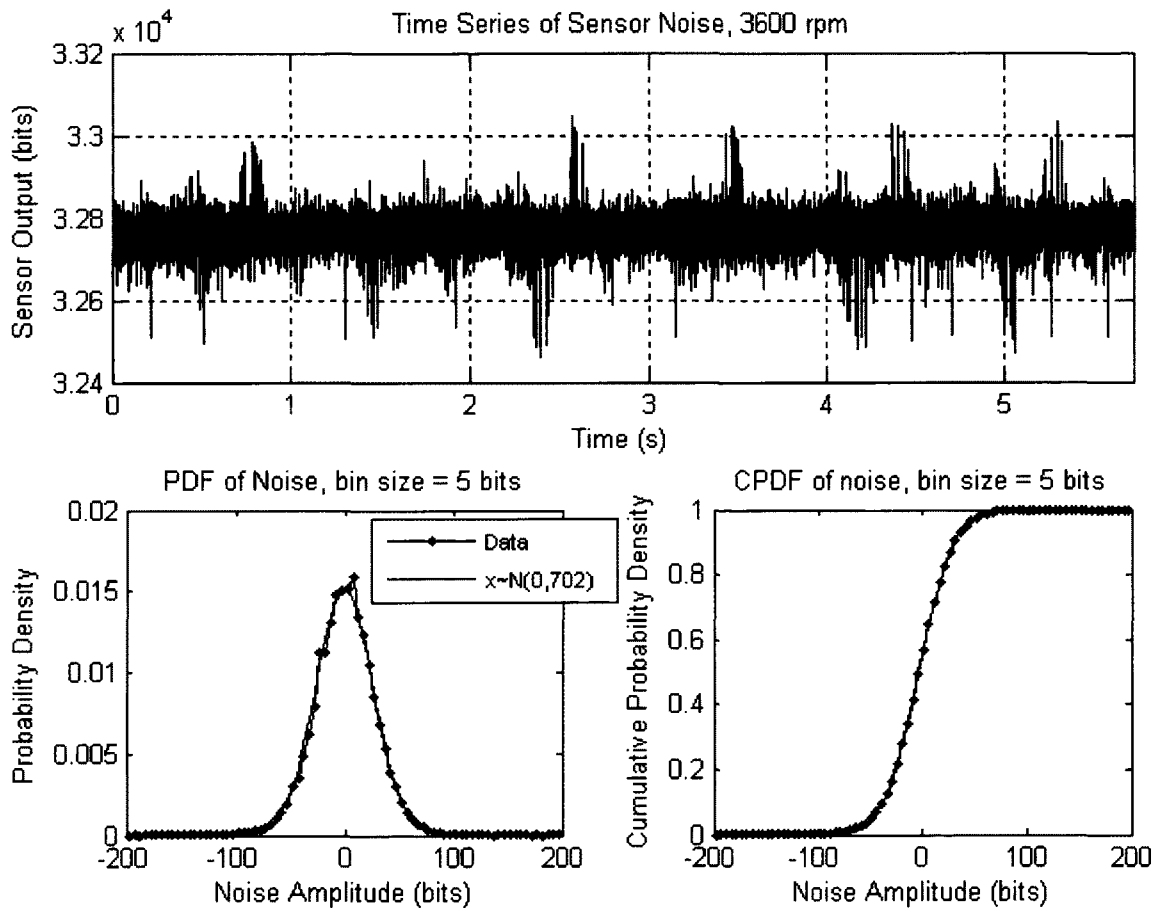


Figure 3.13 - Sensor noise is shown to follow a Gaussian distribution

The noise is observed to follow a Gaussian distribution which means that time-averaging does not bias the result. This is important because all of the static calibration results were obtained by time-averaging the output of the sensor. A standard deviation of 26.5 bits implies that 95 percent of the noise is contained within ± 2 sigma of the mean: Therefore, 95 percent of the

noise is between +/- 53 bits, corresponding to 0.16 percent full scale. Thus, the effective resolution is calculated as:

$$Effective\ Resolution = \frac{2(53\ bits)}{23.5\ bits/N} = 4.51\ N\ (1.01\ lbf) \quad (3.5)$$

Therefore, the effective resolution satisfactorily meets the design specification of 4.5 N (1 lbf) presented in Table 2.1.

3.7 Drift and Sensor Stability

Sensor drift was characterized by recording the no-load sensor output over extended periods of time. The results of this drift study are shown in Figure 3.14. The sensor was powered on at time $t = 0$ and remained in a state of zero stress for the duration of the study. The experiment was conducted at room temperature of approximately 25°C. Measurements were taken at approximately 1 minute intervals:

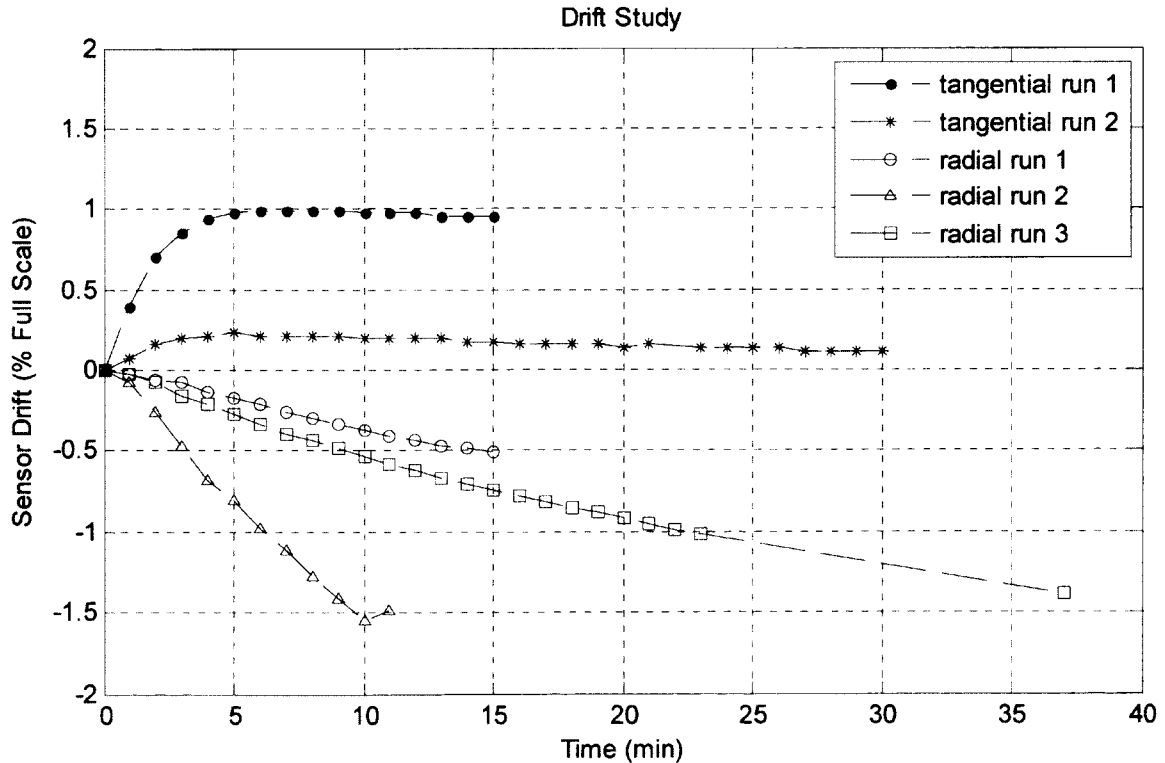


Figure 3.14 - Study of sensor drift on both bridges for extended time records

The behavior of these drift records is quite perplexing: Both of the bridges are conditioned by the same electronics, yet the output of the tangential bridge is seen to level off after approximately 5 minutes while the output of the radial bridge continues to drift downward. The source of this drift is still unknown.

The effects of this drift are almost inconsequential, however, because most milling operations to be observed with this sensor will only last a few minutes. The magnitude of the drift remains small at less than 2 percent full scale and therefore has almost no effect on the span of the sensor. Also, for most geometries of cut, it is easy to determine when the tool is out of the workpiece, and therefore trivial to remove any bias due to drift in post-processing. For future real-time implementation, knowledge of when the tool is out of the workpiece could be used to tare the bias and compensate for drift in a block-adaptive sense.

Another important factor that could potentially affect the DC stability of the sensor is temperature. As stated in Chapter 2, the semi-conductor strain gages are matched to have the same coefficient of thermal expansion. Theoretically, this should eliminate any systematic bias due to temperature effects; however, an experimental characterization has not been completed. Sensitivity to temperature should be investigated as future work.

3.8 Summary

An exciting result that comes from static calibration is the ability to determine the total measurement uncertainty for static loads applied to the sensor. Having characterized the various sensitivities, cross-sensitivities, and noise statistics, it is possible to use a truncated Taylor Series to determine the total measurement uncertainty via propagation of errors. To do so, we express the static measured force as

$$F = \frac{\cos \theta}{l} \cdot S \cdot \varepsilon \quad (3.6)$$

Where θ is the misalignment angle between the gage axis and the axial direction of the tool holder, S is the bridge sensitivity, l is the distance from the cutting insert to the gage location, and ε is the measured strain. Using a truncated Taylor Series, total measurement uncertainty for the force is given by:

$$\Delta F = \sqrt{\sum_{i=1}^N \left(\frac{\partial F}{\partial x_i} \Delta x_i \right)^2} \quad (3.7)$$

$$= \left[\left(\frac{\partial F}{\partial \theta} \Delta \theta \right)^2 + \left(\frac{\partial F}{\partial l} \Delta l \right)^2 + \left(\frac{\partial F}{\partial S} \Delta S \right)^2 + \left(\frac{\partial F}{\partial \varepsilon} \Delta \varepsilon \right)^2 \right]^{1/2}$$

The uncertainty in planar misalignment of the Wheatstone bridge, $\Delta\theta$, is determined from the cross-sensitivity analysis of Chapter 2. The conservative value of 3 degrees is used to obtain static confidence intervals. The uncertainty in lever arm distance, Δl , was obtained by taking 10 measurements from the cutting insert to the bridge location. Student's T-distribution is used to assign an uncertainty to this distance. Uncertainty in the bridge sensitivity, ΔS , is obtained from the 95% confidence interval for linear regression, and is a nonlinear function of applied force, F . Lastly, the uncertainty in the measured strain signal, $\Delta\varepsilon$, is a function of the sensor noise, the bending crosstalk, the torsional crosstalk, and the axial crosstalk. These sources of error vary with the ratio of the radial force to the tangential force, and with the value of force itself. For this reason, a family of curves is used to prescribe confidence intervals for measurements.

The 95% static confidence intervals for the tangential force and radial force are shown in Figure 3.15 and 3.16 respectively. Because the crosstalk changes as a function of the ratio of cutting forces, uncertainty also changes as a function of this ratio. Near the extreme value for worst-case scenario, a 45 N (10 lbf) uncertainty in measured force corresponds to approximately 3 percent uncertainty full scale. Thus, total measurement uncertainty satisfactorily meets the design specification of less than 5 percent full scale as presented in Table 2.1.

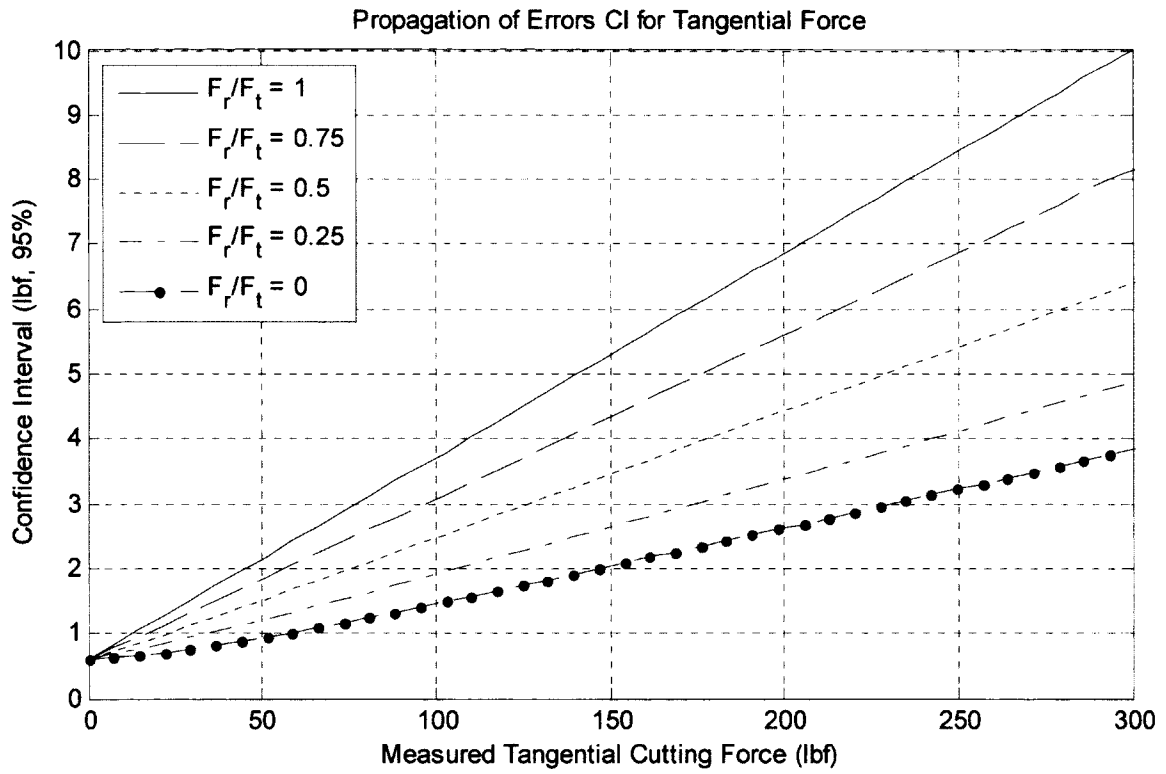


Figure 3.15 - Static confidence intervals for measured tangential force

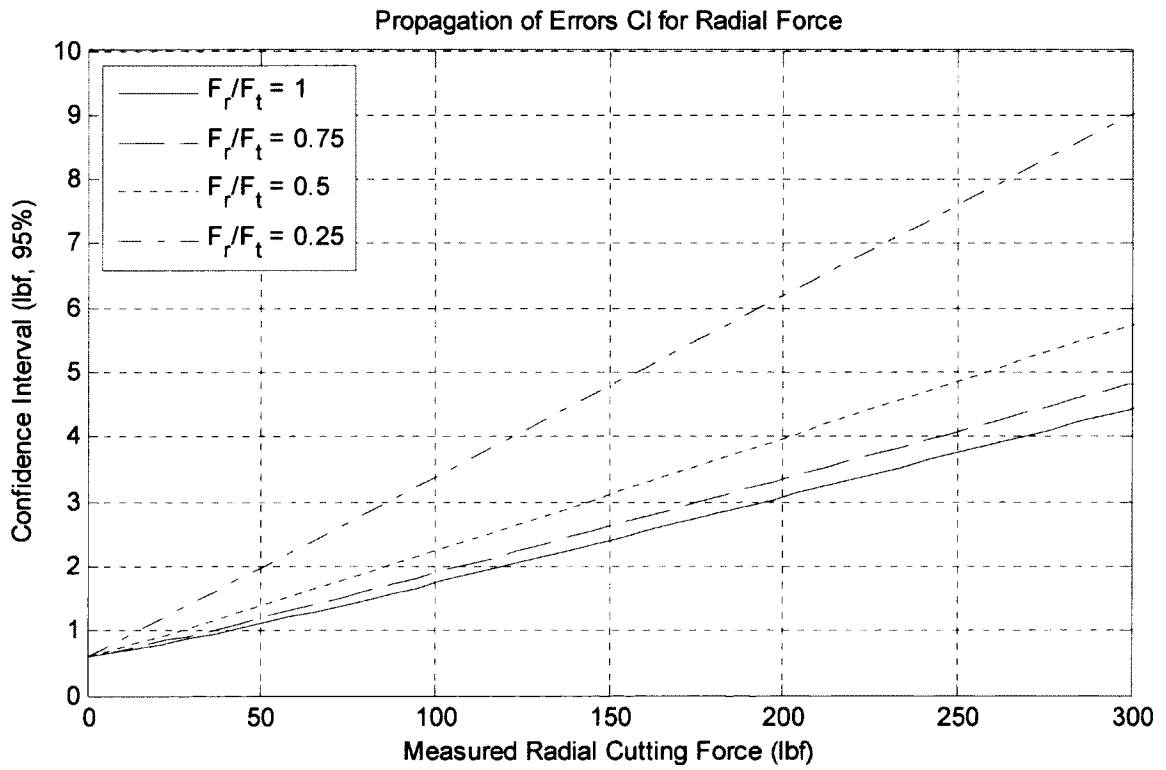


Figure 3.16 - Static confidence intervals for measured radial force

Static calibration has shown that the sensor adequately meets the design specifications outlined in Table 2.1. While bending crosstalk on the radial bridge exceeds the design specification of less than 1% full scale, the total measurement error still satisfies the design constraint of less than 5% full scale. A summary of the significant experimental results is presented below:

Table 3.3 – Summary of experimental results from static calibration

Attribute	Specification	Experimental Result	Specification Met?
Effective Resolution	4.5 N minimum	4.51 N	Yes
Span	1330 N minimum	1395 N	Yes
DC Stability	< 3% full scale	Drift < 2% full scale	Yes
Bending Crosstalk	< 1% full scale	Radial: 0.506 %	Yes
		Tangential: 2.68 %	No
Torsional Crosstalk	< 1% full scale	Radial: 0.37 %	Yes
		Tangential: 0.15 %	Yes
Total Error	< 5% full scale	< 3% full scale	Yes

The ability of Smart Tool v.10 to accurately measure bending forces from a combined loading scenario is truly a significant result. Notwithstanding, we have only considered static loads applied to the sensor. In addition to rejecting unwanted components of strain, the bandwidth of the sensor must allow for accurate force measurement in a dynamic environment. Thus, the dynamic performance of Smart Tool v.10 is evaluated in Chapter 4, and an experimental validation is presented in Chapter 5.

CHAPTER 4

DYNAMIC CHARACTERIZATION

4.1 Introduction

Static calibration shows that the sensor accurately resolves the radial and tangential components of bending strain from a combined loading scenario. Furthermore, for the static case, the static sensitivity directly relates the sensor output to the applied force. For the dynamic case, however, sensor output is no longer proportional to cutting force. Now, the net cutting force is a sum of the inertial forces, spring forces, and damping forces. Therefore, understanding the system's dynamic behavior is critical to interpreting the measured strain signal. This section investigates the dynamics of the Smart Tool in our Fadal 3-axis CNC machine and discusses the corresponding implications to interpretation of the bending strain signal.

4.2 Experimental Determination of the Static Frequency Response Function

A baseline characterization of the open-loop dynamic response of the end-milling system is achieved by performing a “hammer test.” Hammer testing is often the easiest and quickest technique for measuring frequency response functions (FRF's) used in modal analysis [15]. This baseline FRF is a static, non-rotating, non-cutting representation of the milling system with a fixed-free boundary condition. It tells us about the natural modes of vibration which may be excited by the cutting process. The hammer test also establishes a baseline for determining the bandwidth of the sensor.

To determine the static, open-loop response of the milling system, a Parlec 6-inch extended tool holder was mounted in the spindle of our Fadal 3-axis CNC machine. This is the

same tool holder that was used to construct Smart Tool v.10. Hammer testing was not performed on the Smart Tool itself because the shrouding around the electronics prevented us from mounting the accelerometer next to the strain gages. Positioning of the accelerometer in the same location as the strain gages is important to accurately model the FRF. The accelerometer used was an ICP piezo-electric accelerometer with a sensitivity of 10 mV/g. The structure was excited in the x-direction using a Modally Tuned impact hammer (Model 086D05) with a sensitivity of 10 mV/lbf. A schematic of the experimental setup is shown below in Figure 4.1.

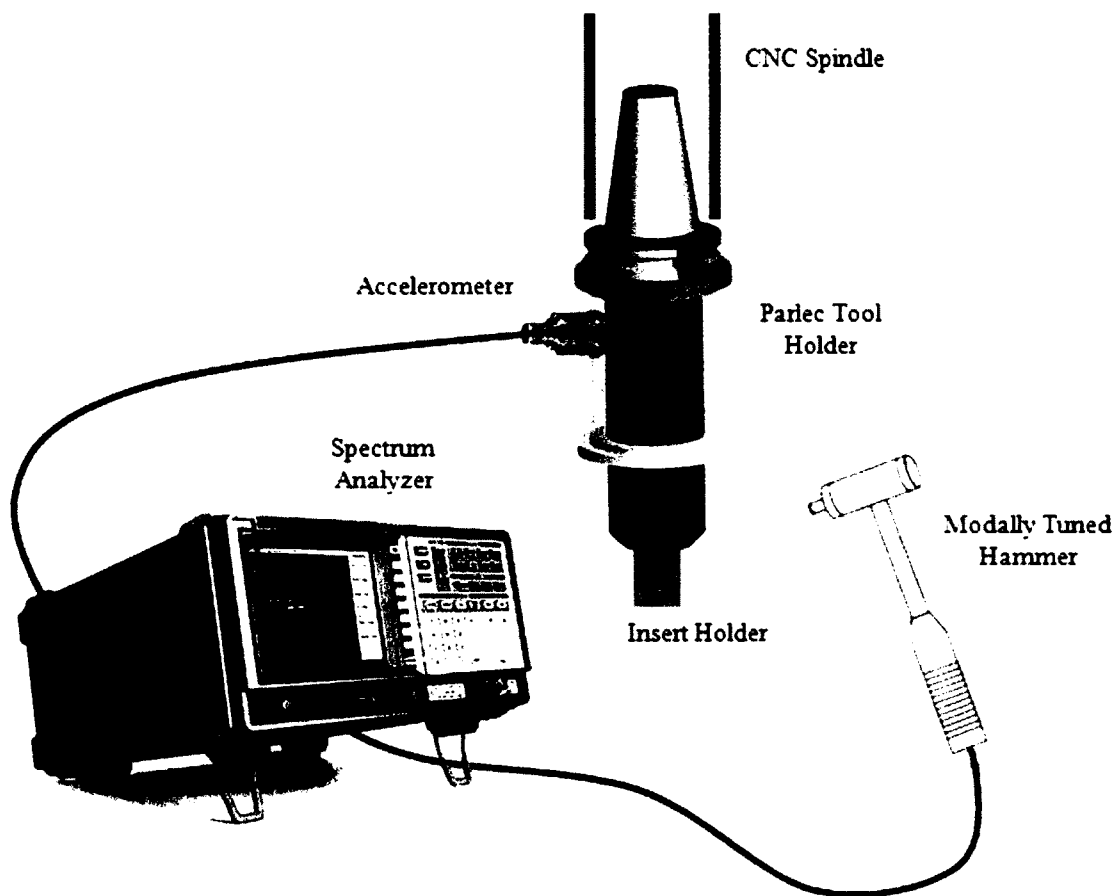


Figure 4.1 - Schematic of the hammer test experimental setup

The experimental protocol for determining the frequency response function was obtained from the literature for our Agilent 35670A spectrum analyzer [15, 16, 17]. A frequency span of 6.4 kHz was used because it was the option that best matched the Smart Tool's Nyquist frequency of 5.12 kHz. Also, an exponential data window with a time constant of 12 ms was used on the

accelerometer response to reduce spectral leakage. Figure 4.2 shows both the measured excitation force and accelerometer response for a typical impact:

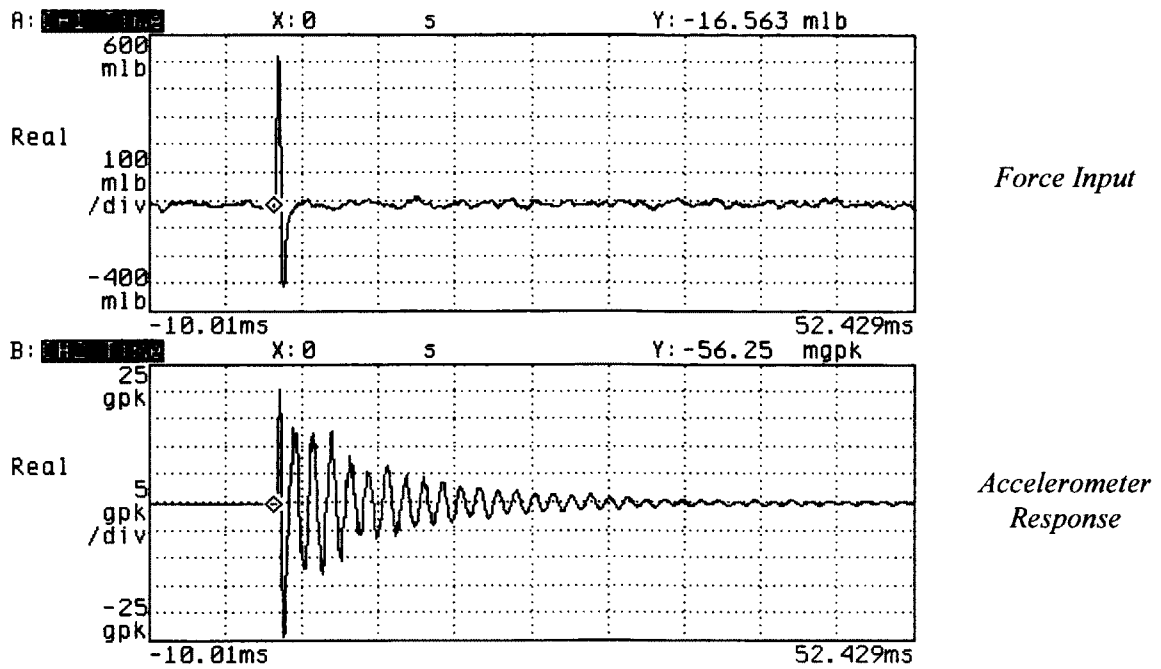


Figure 4.2 - Force input and accelerometer response measured by the spectrum analyzer

Poor Coherence

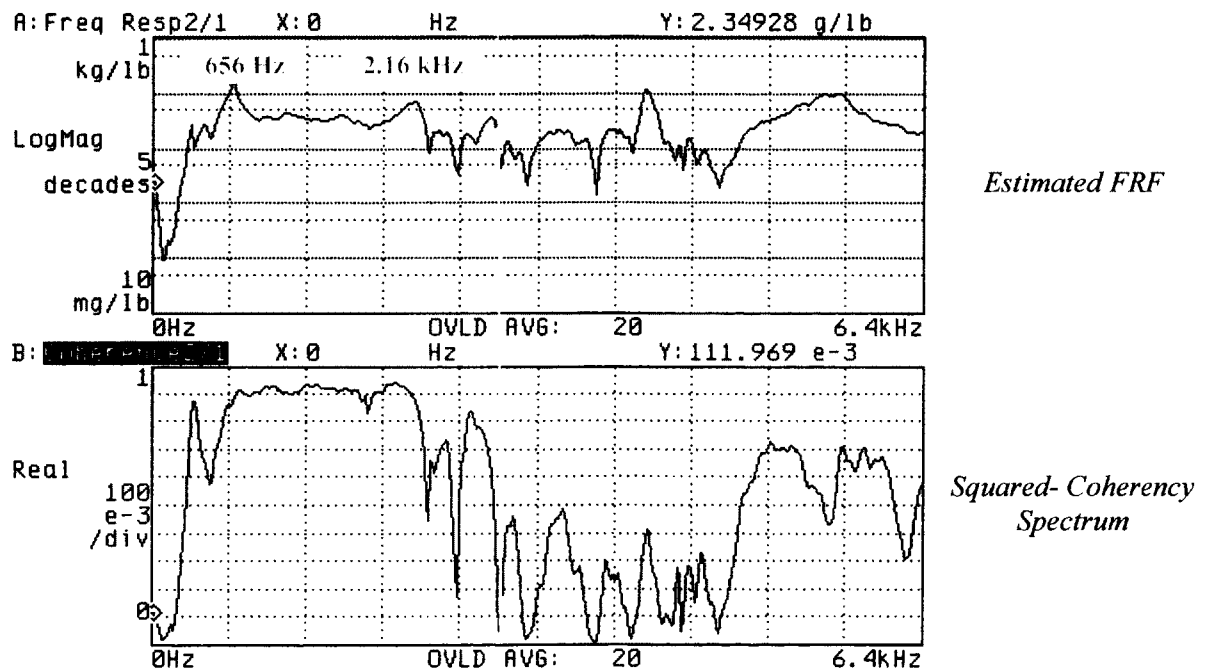


Figure 4.3 - Estimated frequency response function and squared coherency spectrum

The estimated frequency response function and corresponding squared-coherency spectrum are shown in Figure 4.3. The estimated FRF's for twenty hammer impacts were ensemble-averaged to improve confidence in the spectrum. A nylon tip was used on the impact hammer to concentrate energy at low frequencies and to avoid double-taps of the hammer. As a result, the magnitude of the FRF contains no coherent information beyond approximately 2.8 kHz. From this experiment, we observe that the bandwidth of the open-loop system is quite low at approximately 600 Hz. We also observe at least two resonant modes in the open-loop static transfer function. These resonant modes correspond to the spectral peaks seen in the FRF at approximately 656 Hz and 2.16 kHz. Using the curve-fit analysis of the spectrum analyzer, the system poles corresponding to regions of acceptable coherence are $-69.57 \pm j 679.68$ and $-72.017 \pm j 2.117e3$. Because the input did not contain much high-frequency energy, we are unable to accurately identify closed loop poles beyond this range. The experimental FRF and corresponding curve fit are shown below in Figure 4.4.

Poor Coherence

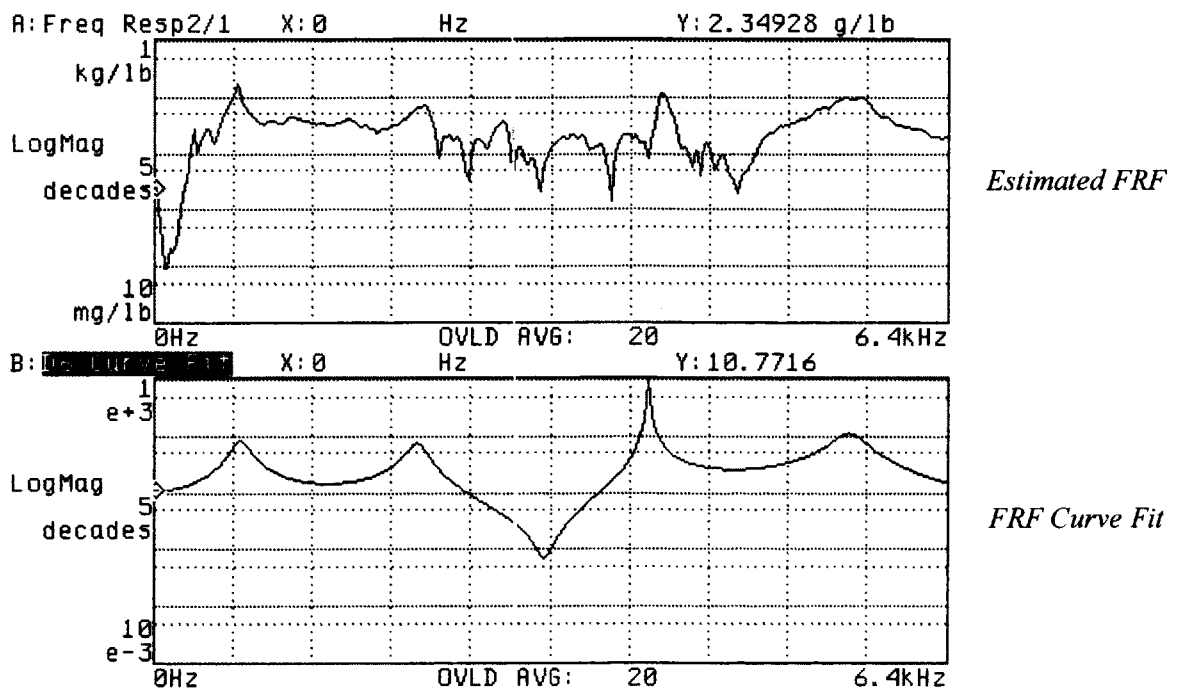


Figure 4.4 - Estimated FRF and corresponding FRF curve fit

The challenge of working with a sensor with such a low bandwidth is manifest by the fact that the system resonances are easily excited by the cutting process. This can become a problem for displacement-based force sensors like our Smart Tool when measured strains are no longer directly proportional to the applied force. How we interpret force measurements corrupted by system dynamics is discussed more completely in chapters 5 and 6. The important result here is that low bending stiffness corresponds to low system bandwidth, thus the time-varying dynamic response of the milling system plays an integral role in accurate force measurement.

4.3 Variation in the Natural Frequency and Damping Ratio with Spindle Speed

When the bandwidth of a sensor exceeds the maximum frequency content of the input signal, the dynamic response of the system is of little importance because the resonant modes are not excited by the input. However, with a bandwidth of approximately 600 Hz, we can expect the system dynamics to play a significant role in accurate force measurement. For this reason, we performed a simple experiment to track how the open-loop dynamic response of the tool-and-spindle system changes with spindle speed.

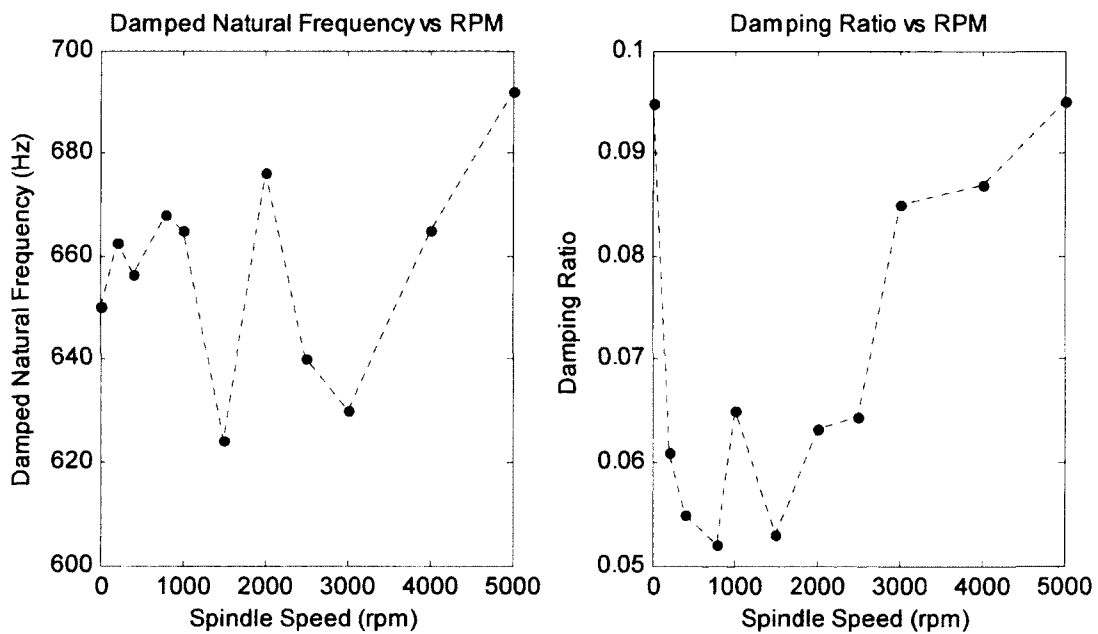


Figure 4.5 - Dynamic variation in the natural frequency and damping ratio

Figure 4.5 shows a preliminary data set investigating the out-of-cut (open-loop) dynamics of our sensor when used with a Fadal 3-axis CNC machine. For each spindle speed, the sensor was excited by tapping the Smart Tool sensor with a hammer. The strain signal of the sensor was analyzed to investigate variation in the natural frequency and damping ratio as a function of spindle speed. The damped natural frequency was determined by calculating the average period of vibration, and the damping ratio was estimated by using a least-squares multivariate regression to the damped response. The data points are all single observations and we acknowledge that further testing should be done to improve statistical confidence. While the natural frequency and damping ratio are seen to change with spindle speed, we do not see a clear deterministic structure to their behavior. Note that the CNC machine experiences a gear change at 2500 RPM; this may affect the boundary conditions of the sensor.

This simple experiment tells us that the dynamic response of the sensor changes with different boundary conditions, and that this variation is somewhat stochastic. This stochastic nature makes it difficult to remove unwanted vibrations through simple linear filtering because the natural frequency moves without any clear deterministic structure. Furthermore, this experiment does not consider the effects of changing the free boundary condition at the tooth. By adding friction at the tool tip, one would expect both the stiffness and the damping to increase. Schmitz et al. [18] have shown that process damping effects are velocity-dependent and that damping increases with spindle speed for constant feedrate. The general problem with time-varying system parameters is that some sort of dynamic system identification becomes necessary in order to compensate for the error in the measurement.

4.4 Boundary Conditions and Dynamic Parameter Identification

The system poles will change during cutting because the sensor's boundary conditions change with parameters like spindle speed, feedrate, radial immersion, axial depth, temperature, tool wear, etc. Since signal processing will be necessary to improve the accuracy of dynamic

force measurements, it is first necessary to develop a modeling tool that tracks changes in the resonant modes of the milling system. Linear Predictive Coding (LPC), formally presented in Chapter 6, conveniently lends itself to perform dynamic parameter estimation. LPC is a least-squares technique that uses the autocorrelation of a signal to model resonant structures in the data. For example, a sixth order auto-regressive model can be used to model the three spectral peaks repeatedly seen in free vibrations of the system, as shown below in Figure 4.6.

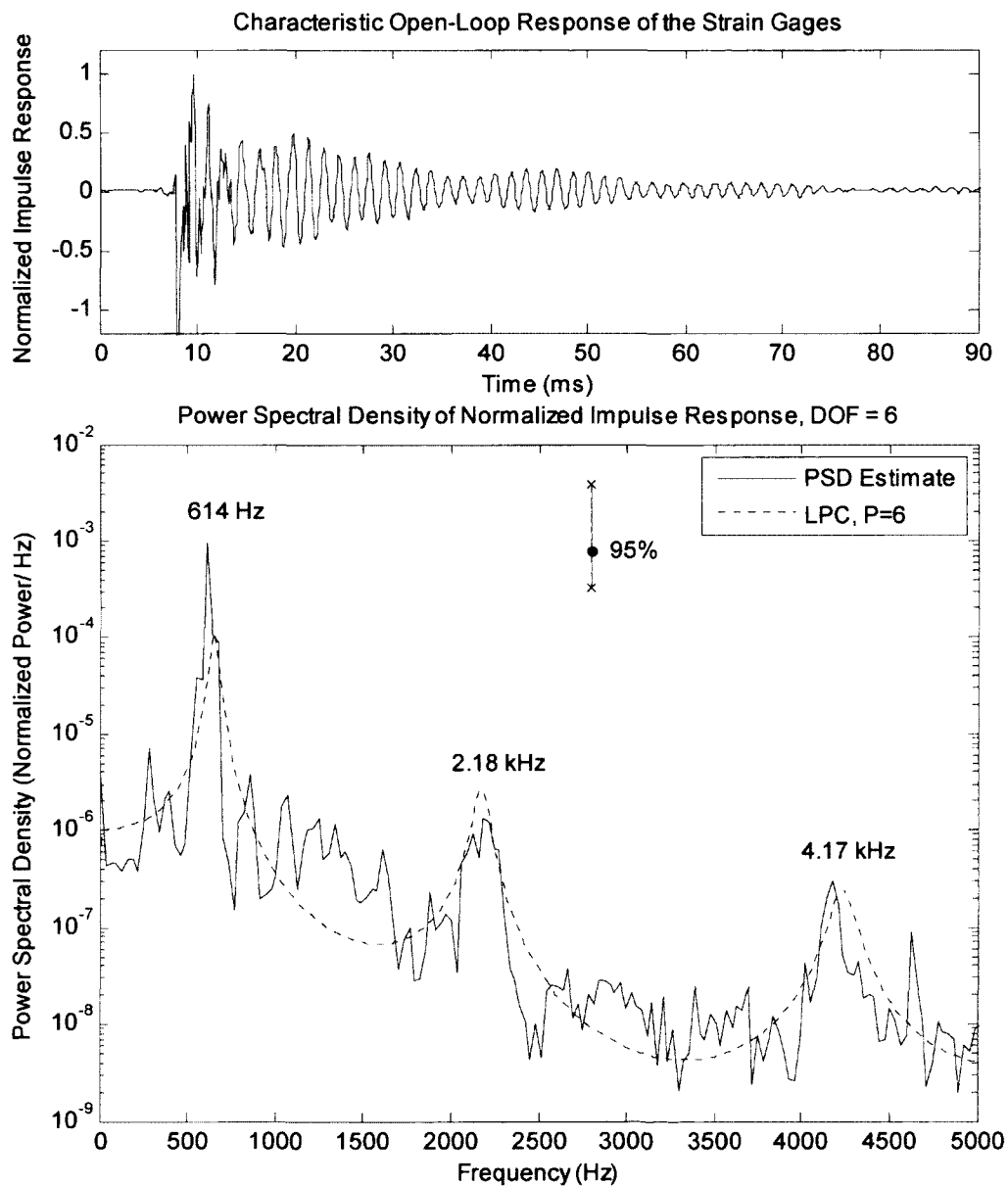


Figure 4.6 - Example of using LPC for system identification; the LPC-based PSD estimate is formed from the auto-regressive model to illustrate the frequency response of the system model

Details of the LPC implementation and a case study of dynamic parameter estimation are presented in Chapter 6. An example of LPC was only mentioned here for completeness, illustrating that we have a technique capable of modeling the resonances of the system. This system model can subsequently be used to remove unwanted dynamics from the measured data.

4.5 Mass Distribution and Systematic Bias

A final note concerning the dynamic performance of Smart Tool v.10 is that uneven mass distribution of the conditioning electronics creates a systematic bias as shown in Figure 4.7.

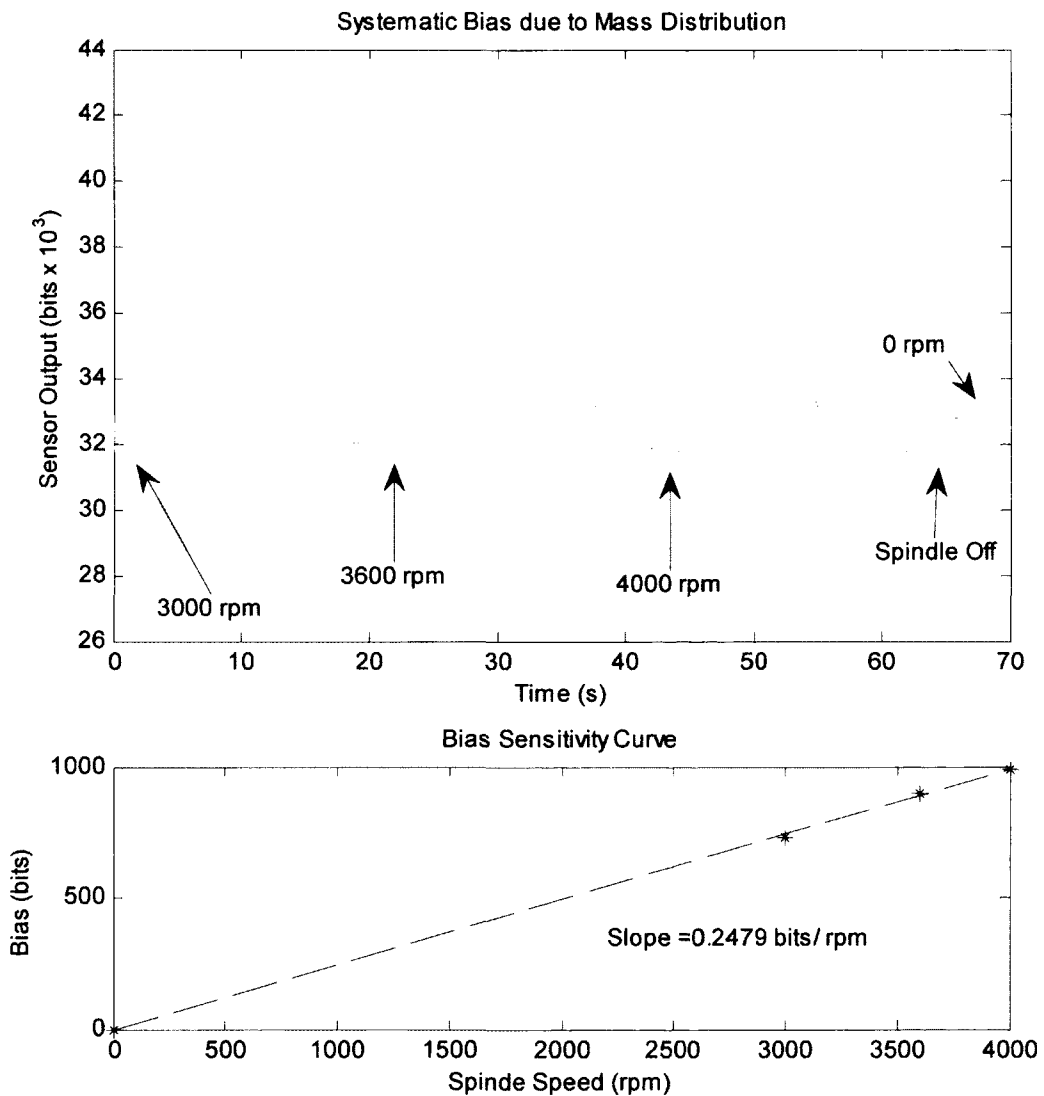


Figure 4.7 - Systematic bias as a result of uneven mass distribution around the tool holder

This bias is not a problem for a single tooth cutter because there is no runout when only one tooth is used. Notwithstanding, the “zero” will need to be re-estimated each time the spindle changes RPM in order to compensate for this bias.

The full-scale reduction in span due to uneven mass distribution is calculated as:

$$\text{Bias Error} = \frac{(0.2479 \text{ bits/rpm})(7500 \text{ rpm})}{32,768 \text{ bits}} \times 100 = 5.67\% \text{ full scale} \quad (4.1)$$

While this dynamic bias does reduce the effective span of the sensor, it is not enough to warrant adding mass to even out the distribution, as increasing mass would further reduce the bandwidth. If future sensor designs use more than one cutting tooth, however, extra care should be taken to evenly distribute mass of the conditioning electronics. If not accounted for, the multi-tooth design will suffer from problems with runout.

4.6 Summary

This chapter shows that the sensor’s bandwidth of approximately 600 Hz will be a problem for accurate force measurement because the frequency content of the applied force is prone to excite the resonant modes of the milling system. To successfully remove unwanted vibrations from the strain signal, some technique must be implemented to perform dynamic system identification. Chapter 6 will show how Linear Predictive Coding can be employed to track the system resonances. A dynamic technique for model identification is necessary because of the stochastic nature observed in the variation of dynamic parameters when the system is subjected to changing boundary conditions.

CHAPTER 5

EXPERIMENTAL VALIDATION

5.1 Introduction

The ability of the Smart Tool to dynamically measure force is evaluated by performing a sensor comparison with a 3-axis bed dynamometer made by Kistler Instrumentation Corporation. This Kistler bed dynamometer is commonly used in research laboratories, thus, its performance provides a benchmark by which to compare the capabilities of our Smart Tool. A series of up-milling operations at 600 rpm are performed to compare net force profiles. Higher speed upmilling is also performed at 3,000 RPM, 3,600 RPM, and 4,000 RPM to compare the accuracy and dynamic characteristics of each instrument. The Smart Tool is shown to provide comparable measurement accuracy and dynamic performance.

5.2 Comparison of Kistler and Smart Tool v.10

Comparison of measurement accuracy and dynamic performance is most readily achieved by comparing the net force profiles of both the Kistler and the Smart Tool. We choose to compare the net force because the Kistler operates in a fixed X-Y coordinate system while the Smart Tool uses the rotating tangential-radial coordinate system of the spindle. Experimental validation is further complicated by the fact that the Smart Tool is currently only capable of measuring a single channel at a time. Therefore, to determine net forces on the Smart Tool, each cutting test must be repeated and cut-to-cut variability must be shown to be negligible.

5.2.1 600 RPM Validation

The ability of the sensor to accurately measure cutting forces for low tooth passing frequency was investigated by performing a series of milling operations at 600 RPM. The experiment consisted of upmilling operations in aluminum at 600 RPM for radial immersions of $\frac{1}{4}$, $\frac{1}{2}$, and $\frac{3}{4}$. For each immersion, the feedrate was adjusted to obtain an average chip thicknesses of 0.0254 mm, 0.0508 mm, 0.0762 mm, and 0.0106 mm (corresponding to 0.001 in, 0.002 in, 0.003 in, and 0.004 in respectively).

To validate the accuracy of our sensor as a force transducer, the output is compared to that of a 3-axis force dynamometer made by Kistler Instrumentation Corporation. Because the Kistler dynamometer measures force in the X-Y coordinate system, and our sensor measures force in the rotating tangential-radial coordinate system, the outputs of the two sensors are compared by looking at the net force profiles. The net cutting force is calculated from the experimental data as:

$$F_{net} = \sqrt{F_x^2 + F_y^2} = \sqrt{F_t^2 + F_r^2} \quad (5.1)$$

Aligning the force profiles for both qualitative and quantitative comparison requires significant post-processing because the signals were sampled at different sampling frequencies and from two different milling operations. To generate all force profiles, data reduction was performed in the following manner (See complete details and MATLAB code in Appendix D):

First, 20 cycles of the Kistler data were overlaid and time-aligned using the unbiased definition of the time-lagged cross-correlation coefficient.

$$\rho_{xy}(\tau) = \frac{R_{xy}(\tau)}{s_x(\tau) \cdot s_y(t + \tau)} \quad (5.2)$$

Where R_{xy} is the unbiased time-lagged cross-covariance, τ is the lag time, and s_x, s_y are the unbiased standard deviations of the random variables X and Y. The correlation coefficient is a measure of the linear dependence between two time series and therefore serves as a good means

of aligning linear-dependent data. The aligned force cycles were then ensemble-averaged to obtain the characteristic net force profile. Next, the two data sets collected from the Smart Tool were time-aligned using the time-lagged cross correlation, and then ensemble-averaged to obtain the characteristic net force profile. Because the Smart Tool and Kistler data sets were sampled at different rates, the Smart Tool (ensemble-averaged net force) was interpolated onto the slower time axis of the Kistler data by means of linear interpolation. Lastly, the net force profiles of the Kistler and Smart Tool were time-aligned using the cross-correlation coefficient as defined in Equation 5.2.

A flow chart of the data processing required to compare net force profiles is shown below in Figure 5.1:

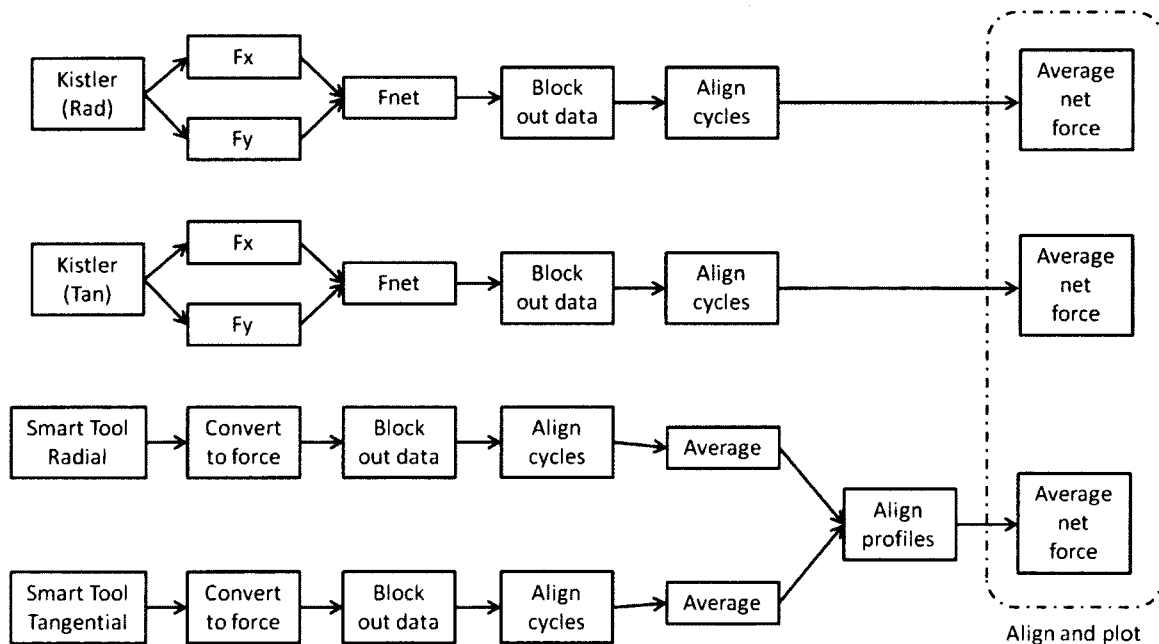


Figure 5.1 - Flow chart of data processing required for net force comparison

The resulting net force profiles for the Kistler and Smart Tool v.10 are shown on the next page in Figure 5.2.

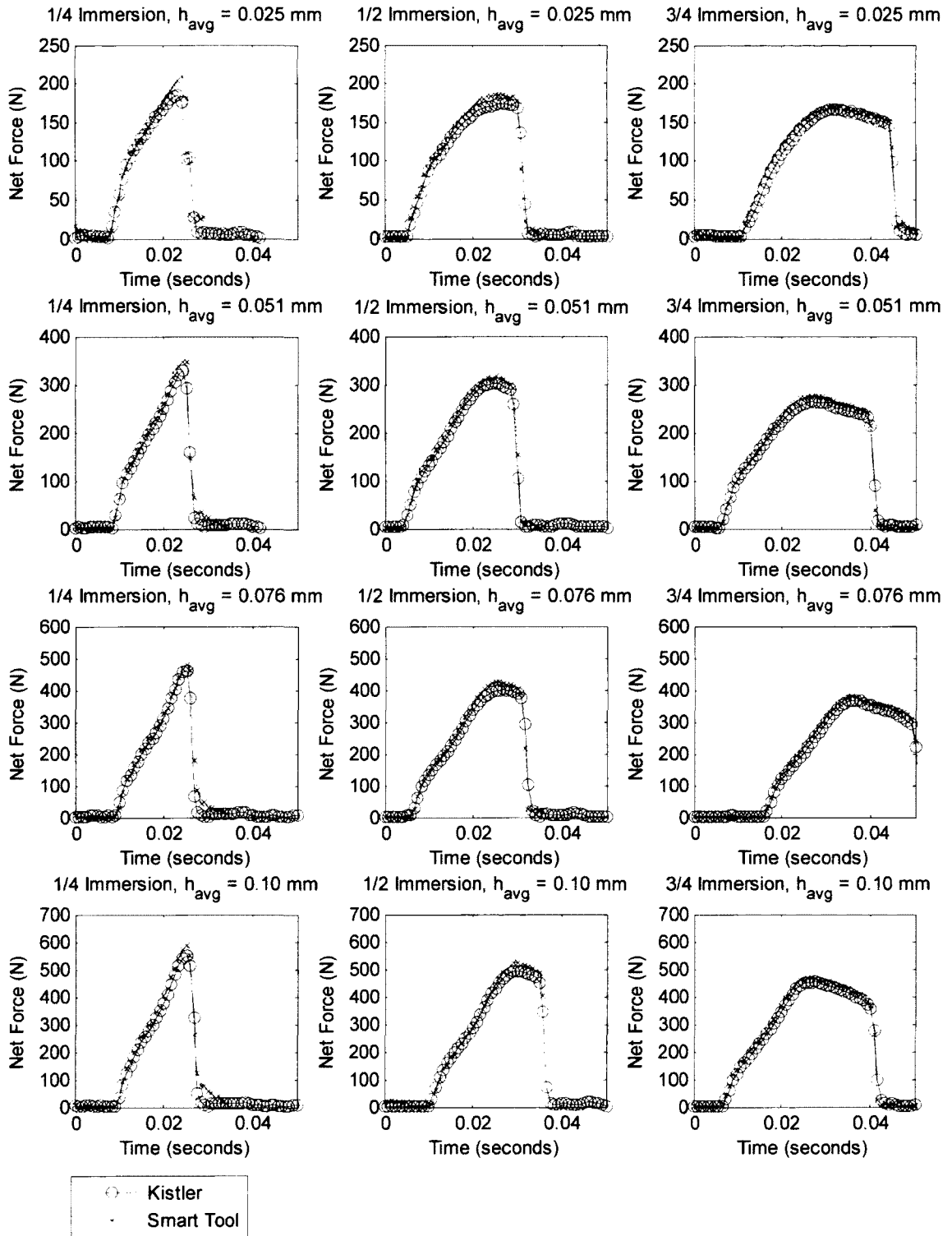


Figure 5.2 - Comparison of net force profiles between Smart Tool v.10 and the Kistler 3-axis force dynamometer; aluminum upmilling at 600 RPM, no coolant

We observe very good agreement between the net force profiles as measured by the Kistler dynamometer and our Smart Tool sensor. In all cases, the peak force measured by the Smart Tool is slightly larger than that measured by the Kistler. For such low toothpassing frequency (10 Hz), tool vibration is negligible and the two sensors provide nearly identical force measurements. Only one of the two Kistler data sets is shown; such good agreement between the Kistler and Smart Tool profiles shows that cut-to-cut variability is negligible.

5.2.2 3000 RPM, 3600 RPM, and 4000 RPM Validation

In addition to performing a validation for the very low tooth-passing frequency of 10 Hz, the same upmilling experiment was performed for toothpassing frequencies of 50 Hz, 60 Hz, and 66.67 Hz corresponding to 3,000 RPM, 3,600 RPM, and 4,000 RPM respectively.

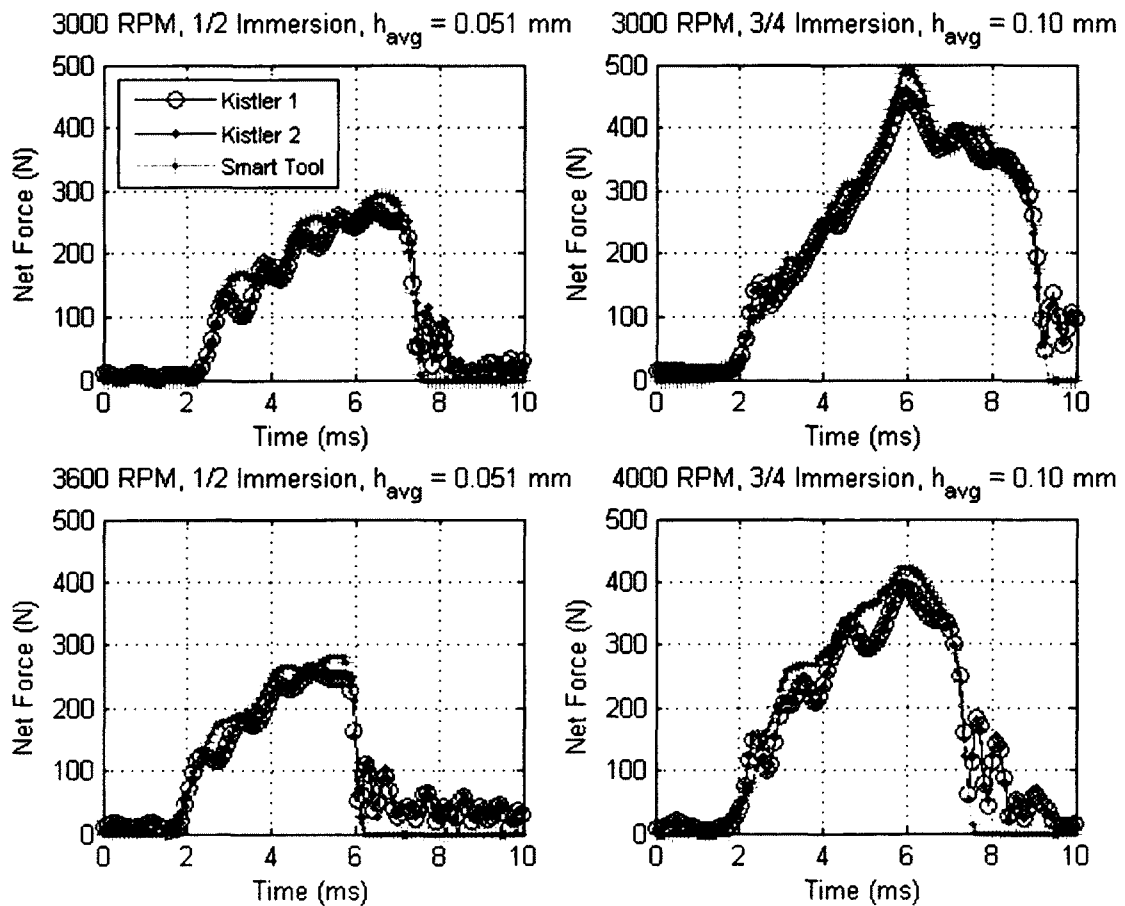


Figure 5.3 - Comparison of net force profiles between Smart Tool and Kistler 3-axis force dynamometer; aluminum upmilling at 3000, 3600, 4000 RPM, no coolant

Figure 5.3 shows four of the 36 unique cutting conditions encountered in this experiment (three spindle speeds \times three immersions \times four feedrates = 36 unique cutting conditions). The data reduction procedure for this experiment is identical to that explained above for the 600 RPM validation experiment. Plots of all net force profiles can be found at in Appendix D.

Because the Kistler and Smart Tool profiles are in less agreement here than they were for the 600 RPM case, both Kistler data sets are plotted in Figure 5.3. “Kistler1” corresponds to the milling operation with the Smart Tool transmitting tangential strain data, and “Kistler2” corresponds to the milling operation with the Smart Tool transmitting radial strain data. The agreement between the Kistler profiles, coming from two different milling operations, validates the approach of collecting tangential strain and radial strain data independently.

These four particular cutting conditions were chosen to include here because they best illustrate the similarities and differences between our Smart Tool sensor and the Kistler 3-axis dynamometer. Observe that the Kistler profiles and Smart Tool profiles are in general agreement. Both sensors yield similar peak force measurements, and both sensors agree with where tooth engagement begins and ends. This data set is much more interesting than the 600 RPM data set, however, because neither sensor is now accurately measuring the instantaneous milling force! Since both sensors are displacement-based transducers, artifacts of their dynamic behavior corrupt the force measurement.

Recall that the damped natural frequency of a 2nd order system is given by

$$\omega_d = \sqrt{1 - \zeta^2} \cdot \omega_n = \sqrt{1 - \zeta^2} \cdot \sqrt{k/m} \quad (5.3)$$

The damped natural frequency of the Kistler is approximately 950 Hz (depending on the mass of the workpiece), and the damped natural frequency of the Smart Tool is approximately 650 Hz in our Fadal 3-axis CNC machine. The effects of their distinct natural frequencies are best illustrated in the left-side plots of Figure 5.3 where the measurement signals are seen to oscillate about the static chip thickness.

In reality, the net force profile should look much more like the static chip thickness than either sensor indicates. For a stable cut with tool vibration, variation in chip thickness remains in-phase with each period of oscillation. We observe that the force profile does look like the static chip thickness in Figure 5.2 because vibrations are small.

Figure 5.4 from Altintas [1] illustrates the system dynamics relating tool deflection to instantaneous chip thickness.

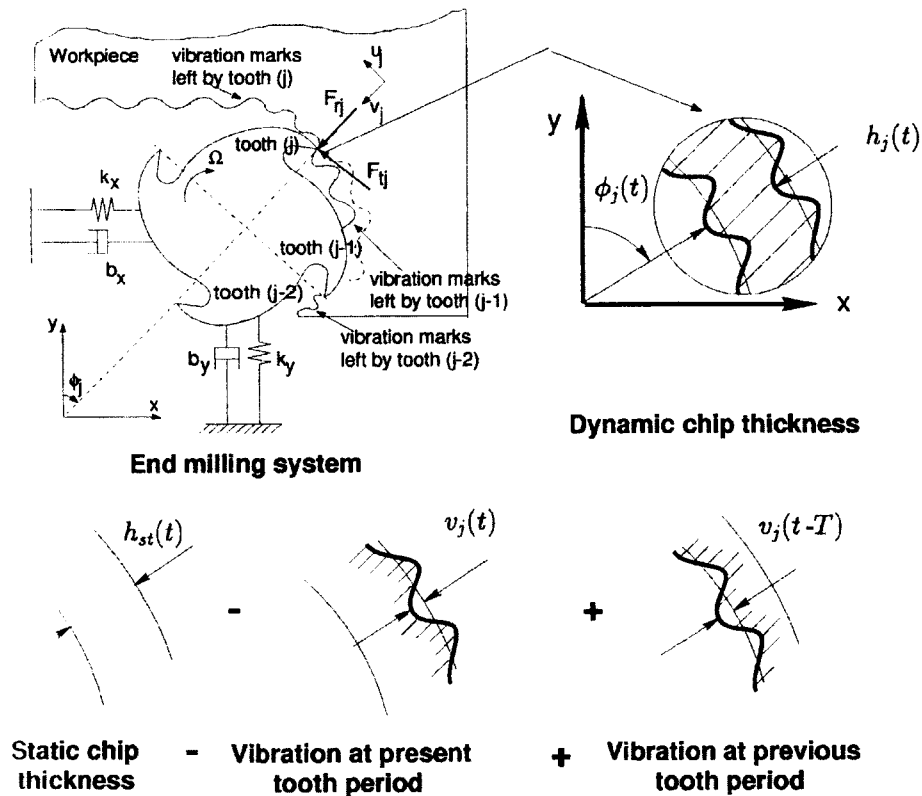


Figure 5.4 - Dynamic chip load model [1]

As the tool vibrates, a surface waviness is created as a result of tool deflection. If this vibration remains in perfect phase with itself for each tooth pass, the dynamic chip thickness is identically equal to the static chip thickness. Variation in the magnitude and phase of tool deflection, however, changes the profile of the dynamic chip thickness. Large changes in chip thickness can lead to instability and chatter which can cause damage to both the tool and to the workpiece.

Since the tool-and-spindle system is approximately an order of magnitude less stiff than the Kistler-and-workpiece system, both sensors exhibit the same general behavior with unique artifacts arising from their different ringing frequencies. The most important characteristic difference between the two sensors is that ringing in the Smart Tool output signal actually tells us something *useful* about the cutting process. Because most of the system compliance is in the tool-and spindle-system, the ringing in the Smart Tool signal provides important information about tool displacement, whereas ringing in the Kistler signal is simply an artifact of that sensor. While too much tool vibration may periodically render the Smart Tool's force measurement inaccurate, it is at these times where spectral analysis could be employed to identify chatter frequencies in search of more stable cutting conditions.

5.3 Summary

Experimental validation shows that Smart Tool v.10 performs with similar accuracy and dynamic performance as the Kistler 3-axis bed dynamometer. However, the Smart Tool's component cost is less than 700 dollars (consumer cost unknown) whereas the Kistler is sold for approximately 35,000 dollars. Peak force measurements have been shown to match those measured with the Kistler: While dynamic effects may make instantaneous force calibration difficult, the unfiltered signal may still be useful in a real-time quality controller when the process objective is to avoid tool breakage or maintain a certain amount of tool deflection. Both of these processes can be controlled by knowledge of the peak force. Notwithstanding, a bandwidth of approximately 600 Hz does degrade the quality of measurements, and there are several signal processing techniques that can be implemented to improve measurements of force. This is the focus of Chapter 6.

CHAPTER 6

APPLICATIONS IN SIGNAL PROCESSING

6.1 Introduction

The ideal measurement system has a system bandwidth that exceeds the maximum frequency content of all expected inputs, thereby making measurements proportional to the input sequence by the static sensitivity. This concept is easily visualized by considering the dynamic performance of an arbitrary measurement system as shown in Figure 6.2.

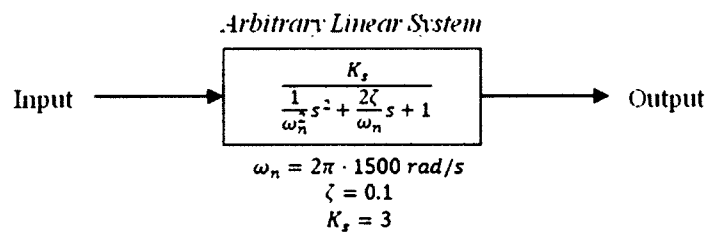


Figure 6.1 - Transfer function of an arbitrary linear system representing a measurement system

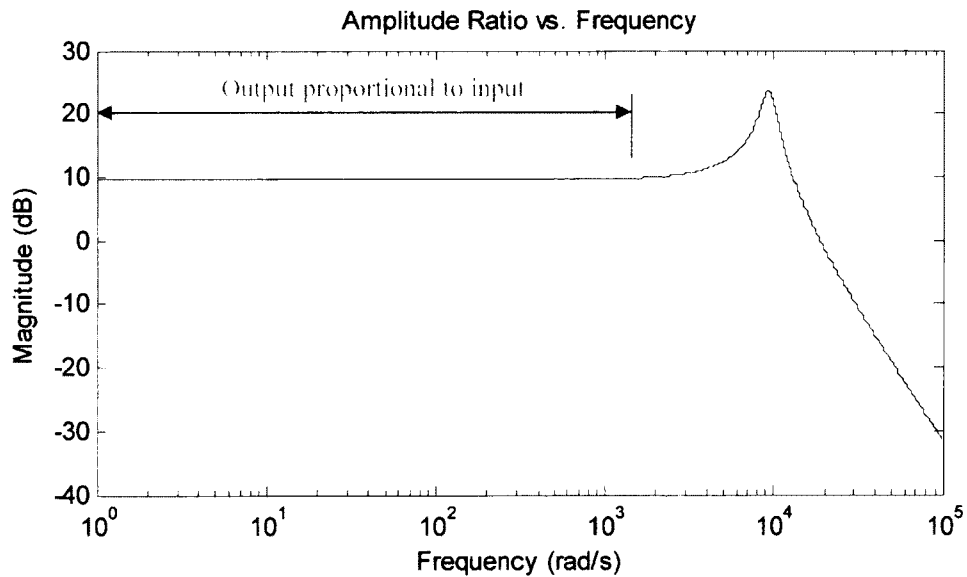


Figure 6.2 - Amplitude ratio of the arbitrary measurement system

In such situations when the system bandwidth exceeds the maximum frequency content of the ensemble of input signals, no signal processing is required to obtain good estimates of those input signals – one simply uses the static sensitivity to scale the measured output.

We face a more challenging measurement situation when the frequency content of the input sequence overlaps with portions of the amplitude ratio that vary in gain. This concept is illustrated in Figure 6.3.

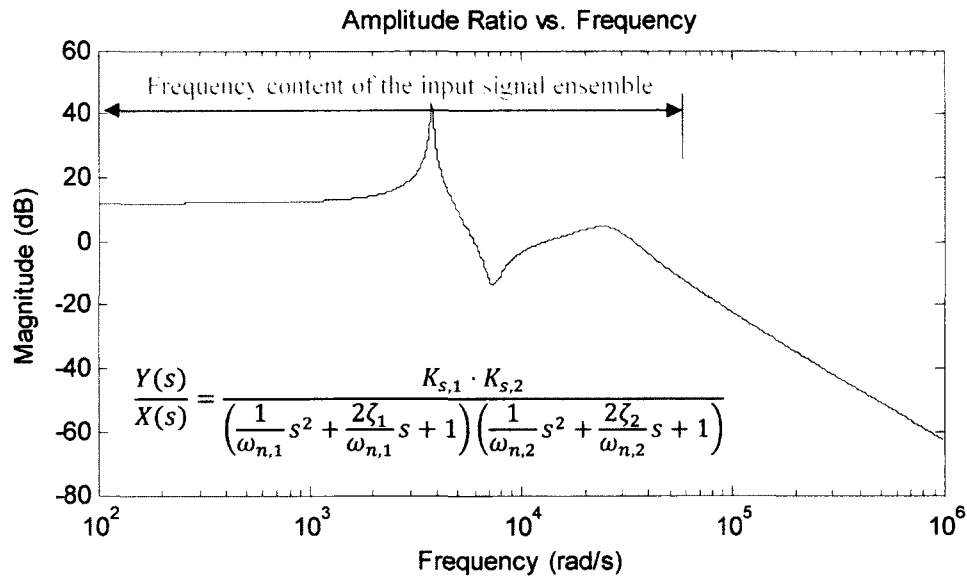


Figure 6.3 - A measurement system whose dynamic parameters affect the accuracy of the measurement

In such a situation where the frequency content of the input signal overlaps with non-constant portions of the amplitude ratio, the system dynamics of the measurement system corrupt the measurement by introducing phase distortion and amplitude modulation.

The mathematical relationship between the input and output signals for a linear causal system can be easily described by means of a convolution integral:

$$y(t) = x(t) * h(t) = \int_0^t x(\tau) \cdot h(t - \tau) d\tau \quad (6.1)$$

Where x is the input signal, y is the output signal, and h is the impulse response of the linear causal system. The problem we now face is *how do we decouple the true measurement from the*

system dynamics? In general, this is a very difficult problem. Perfect recovery of the input signal using a linear equalization filter requires that the linear model of the channel distortion is minimum phase (no zeros outside the unit circle) and that the signal to noise ratio is large. For these reasons, successfully decoupling the system dynamics from the measurement is non-trivial.

Some of the important questions include:

1. Is the system linear or nonlinear?
2. If linear, what is the appropriate model structure?
3. How well does the model fit the data?
4. What is the variance of the modeling error, and how does the model prediction error propagate through estimates of the measurement?
5. Do the dynamic parameters vary with time?
6. Is the estimation algorithm fast enough to implement in real time, or are we limited to post-processing techniques?

These questions guide the development of our signal processing techniques. In comparing estimators, the fundamental parameters used to evaluate any signal processing technique include the quality of the result, the robustness of its application, and its computational efficiency.

6.1.2 Chapter Overview

This chapter presents several techniques in signal processing that have applications in determining how to best interpret the bending strain signal of the Smart Tool when vibrations become significant. Linear Prediction is shown to be a useful tool for model identification, and a simple algorithm for enhanced chatter frequency detection is presented. With model estimates of the linear dynamical system, we then introduce the general problem of state estimation and look at optimal filtering techniques to reconstruct estimates of the measurement from the observation sequence. Sub-optimal filtering techniques are also investigated because of their computational efficiency.

6.1.1 General Approach to Interpreting the Bending Strain Signal

As explained above, the measured strain signal is the result of convolving the applied force with the (time-varying) impulse response of the milling system. Because it is impossible to perfectly model the time-varying system dynamics, we will never be able to perfectly recover the instantaneous cutting force. Our general approach to obtaining useful information from the sensor is conceived by considering a simplified block diagram of the milling system dynamics.

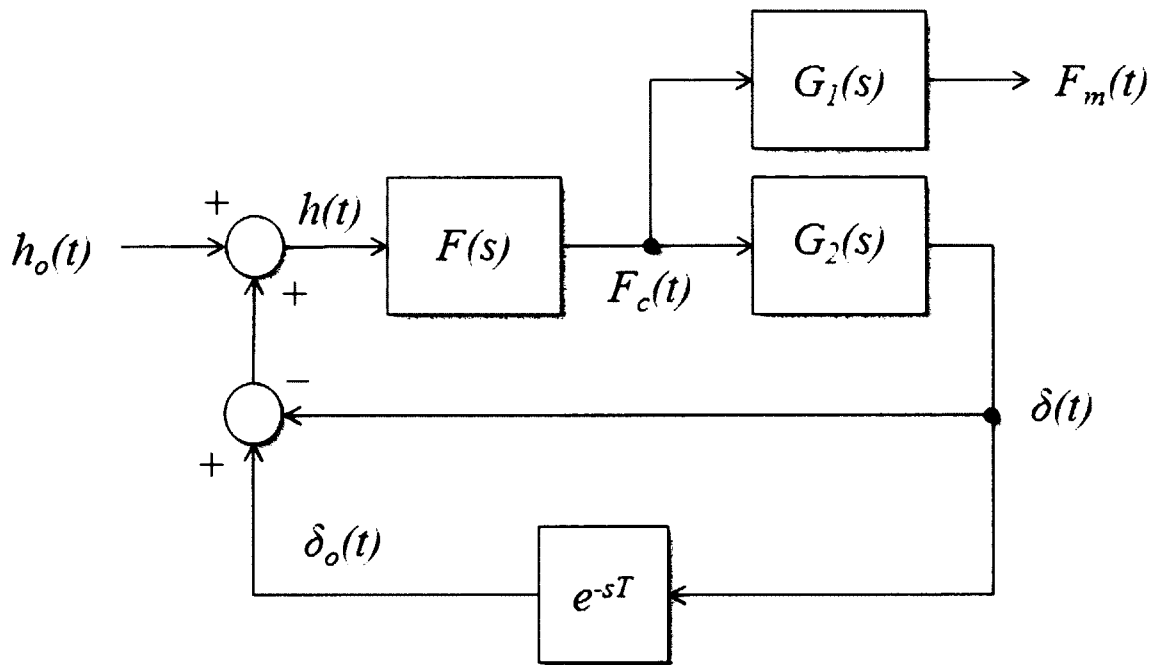


Figure 6.4 - Generalized block diagram of the end-milling process

In Figure 6.4, the measured force, $F_m(t)$, is obtained by multiplying the bending strain by the static sensitivity of the system. This bending strain signal is produced as a result of multiplying the sensor compliance, $G_1(s)$, by the applied cutting force, $F_c(t)$. Here, the sensor compliance, $G_1(s)$, and the end-milling system compliance, $G_2(s)$, are very similar: The sensor compliance starts at the fixed end of the spindle and relates strain on the tool holder to applied force. The system compliance also starts at the fixed end of the spindle and relates tool deflection to applied force. The frequency response of $G_1(s)$ and $G_2(s)$ are therefore nearly identical.

The challenge of recovering F_c from F_m arises from the fact that the system has *memory* because of the delay block. In this sense, we are measuring a deflection component resulting from the system compliance which is subsequently *fed back into the system* to affect the deflection a period later! This is quite different from most applications in channel equalization where the noise (i.e. distortion) is completely independent from the information in the signal. In our application, the compliance distorts the measurement, and the resulting tool deflection is coupled with the next value of the applied force we wish to measure.

The creative solution to obtaining useful force information is hidden in the composition of the applied cutting force:

$$F_c(t) = (\text{force from static chip thickness}) + (\text{force from dynamic chip thickness})$$

Instead of trying to estimate the applied cutting force directly, it is much easier to implement a filtering method that artificially extends the bandwidth of the sensor, thus allowing us to break the actual cutting force into two components:

1. An infinitely-stiff measurement corresponding to the force produced by the static chip thickness
2. A residual measurement containing all of the information pertinent to the system compliance

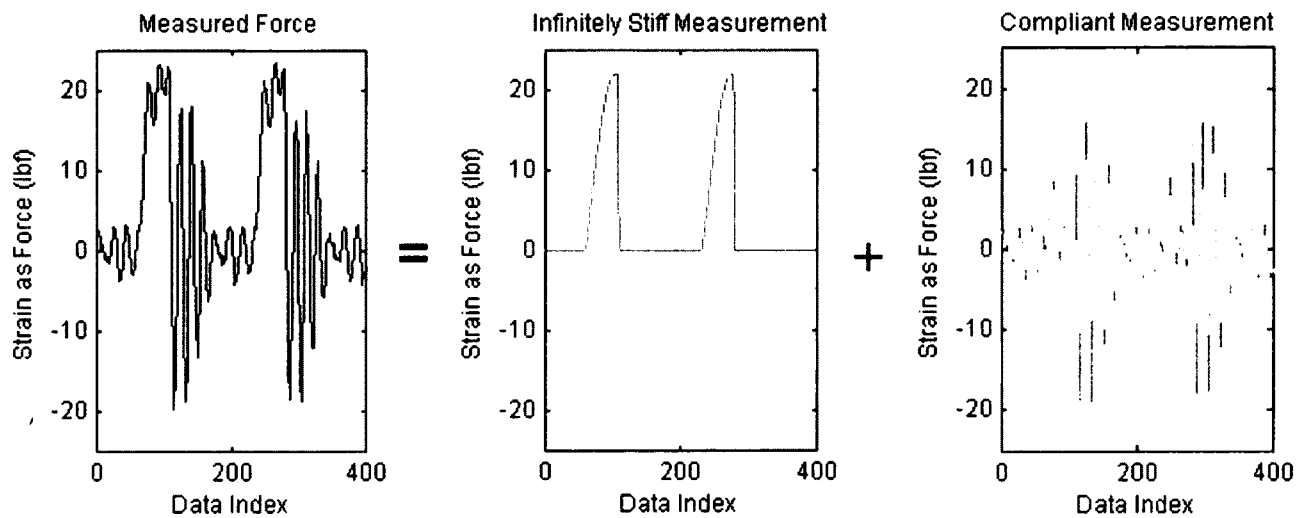


Figure 6.5 - Approach to interpreting the bending strain signal

While the ideal sensor with a very high bandwidth would be capable of measuring the actual cutting force as a result of the instantaneous chip thickness, we can still make meaningful measurements with our Smart Tool by interpreting the strain signal in the following way:

We can first use a filtering algorithm to artificially extend the bandwidth of the sensor by approximating the force that is produced by the static chip thickness in the absence of tool vibration. This is a reasonable thing to do because tool vibrations remain relatively in-phase for stable cuts, thus the applied force would look a lot like the static profile. This filter will produce a measurement that is better suited for online calibration of the cutting coefficients than the measured strain signal.

Next, this “static force estimate” is subtracted from the measured signal to produce the residual measurement that contains all information in the signal pertinent to system compliance. Spectral analysis on this compliant measurement could be used to inform the CNC controller about the stability of the milling operation.

It is important to note that approximating the force due to the static chip thickness (i.e. artificially extending the bandwidth) ignores all information pertinent to tool vibration and system compliance; this is not an unreasonable approach however, because when variation in the dynamic chip thickness is large, it is more important to find stable cutting conditions than it is to accurately resolve the cutting force.

Our ability to break down the signal into static and dynamic components is unique to the Smart Tool because vibrations in the Smart Tool’s strain measurement are largely indicative of tool deflection. This same method of signal conditioning would be ill-suited for the Kistler dynamometer because the signal would tell us mostly about *workpiece* deflection (which is typically a much smaller component of the overall system compliance than tool deflection).

The remainder of this Chapter focuses on developing tools specifically aimed at artificially extending the bandwidth of the sensor and at modeling and interpreting system compliance.

6.2 Linear Predictive Coding for Dynamic Parameter Estimation

Linear Predictive Coding (LPC) provides a robust method for tracking both the open-loop and closed-loop resonances of the spindle-tool-and-workpiece system. Tracking these resonant modes is a necessary part of model building to successfully implement model-based filtering. This is accomplished by specifying the order of a rational polynomial to describe the linear system dynamics, and solving for the best model coefficients. The model coefficients of this rational polynomial are chosen by minimizing the prediction error of the LPC filter in the least squares sense.

6.2.1 Problem Formulation

The objective of LPC in this application is to estimate the pole locations of the dominant resonant modes of the system. These poles can then be analyzed to determine the corresponding system dynamics and subsequently used to develop a good filter for channel equalization.

We begin by prescribing a filter structure that will be used to model the behavior of the bending strain decay profile. Because system compliance is well modeled by resonant structures in the frequency domain, the auto-regressive model of Linear Prediction is well-suited for this application. The filter structure of an auto-regressive (AR) model is given by:

$$H(z) = \frac{1}{A(z)} = \left(1 - \sum_{k=1}^P a_k z^{-k} \right)^{-1}, \quad P = 4 \quad (6.2)$$

In this case, the linear filter is an all-pole filter of order P . Figure 6.6 shows the block diagram used to produce our model-based estimate of the system compliance.

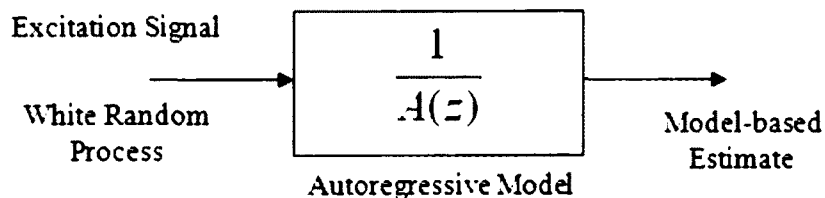


Figure 6.6 - Block diagram of the autoregressive linear prediction model

In model-based spectral estimation, it is assumed that the measured signal can be modeled as the output of a linear time-invariant system excited by a random (i.e. flat-spectrum) input sequence. This assumption that the input has a flat spectrum implies that the power spectrum is shaped entirely by the frequency response of the model [8]. We need this assumption to make the problem mathematically tractable. Of course, a flat power spectrum implies that the driving process has infinite variance, so we must think of the input as band-limited white noise (i.e. finite variance). To validate the AR system model, we can formulate the problem from the opposite direction. Consider the inverse model used to filter the measured signal. If the resulting process is white, it means that the AR model captures all of the important information about the system.

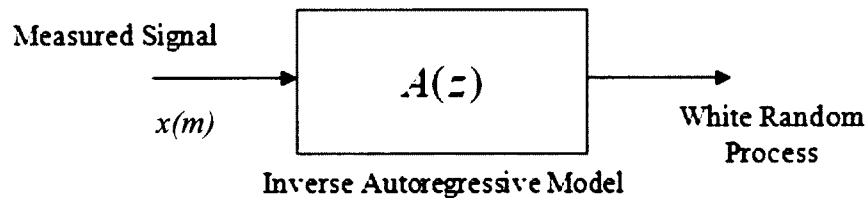


Figure 6.7 - Inverse "spectral whitening" LPC model

Thus, to validate the LPC method for model identification, we look at the model prediction error, which is given by the difference between the measured signal and the model-based estimate:

$$\begin{aligned}
 e(m) &= x(m) - \hat{x}(m) \\
 &= x(m) - \sum_{k=1}^p a_k x(m-k)
 \end{aligned}
 \tag{6.3}$$

If the error sequence is white, then the autoregressive model completely describes the input-output relationship of the system. To test for "whiteness", we can look at the correlation of the residuals. If the error signal is truly white, then it should be uncorrelated with itself for all lags greater than zero. This method of model checking is widely used by Box and Jenkins [10] for the general class of ARMA (Auto-Regressive Moving-Average) models.

An example of how this technique can be used to validate a linear prediction model is shown below in Figure 6.8. Because the error sequence is well decorrelated with itself for all lags greater than zero, this example model adequately captures the behavior of the measured signal.

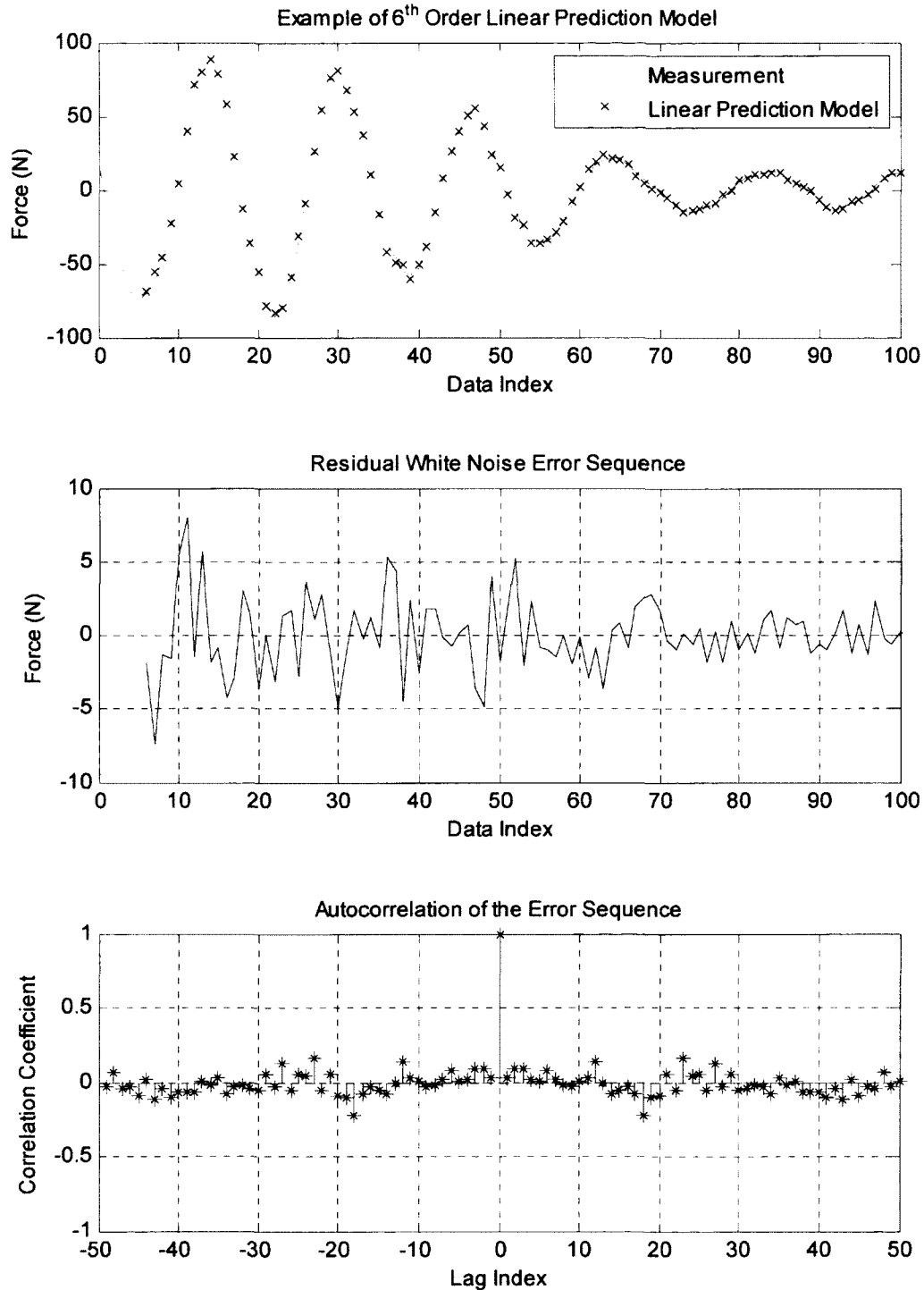


Figure 6.8 – Model checking is performed by looking at the auto-correlation of the residuals

6.2.1 Least Mean Square Error Predictor

The “best” predictor coefficients are obtained by minimizing the Mean Square Error (MSE) defined as:

$$\begin{aligned}
 \text{MSE} &= E[e^T(m)e(m)] \\
 &= E\left[\left(x(m) - \sum_{k=1}^p a_k x(m-k)\right)^2\right] \\
 &= E[x^2(m)] - 2 \sum_{k=1}^p a_k E[x(m)x(m-k)] + \sum_{k=1}^p a_k \sum_{j=1}^p a_j E[x(m-k)x(m-j)] \\
 &= r_{xx}(0) - 2\mathbf{r}_{xx}^T \mathbf{a} + \mathbf{a}^T \mathbf{R}_{xx} \mathbf{a} \tag{6.4}
 \end{aligned}$$

Where $\mathbf{R}_{xx} = E[\mathbf{x}\mathbf{x}^T]$ is the autocorrelation matrix of the input vector, $\mathbf{r}_{xx} = E[x(m)\mathbf{x}]$ is the autocorrelation vector, and $\mathbf{a}^T = [a_1, a_2, \dots, a_p]$ is the predictor coefficient vector. The gradient of the mean square prediction error with respect to the predictor coefficient vector is thus

$$\frac{\partial}{\partial \mathbf{a}} E[e^2(m)] = -2\mathbf{r}_{xx}^T + 2\mathbf{a}^T \mathbf{R}_{xx} \tag{6.5}$$

The least mean square error solution, obtained by setting the gradient equal to zero, is given by

$$\mathbf{a} = \mathbf{R}_{xx}^{-1} \mathbf{r}_{xx} \tag{6.6}$$

This is the typical Yule-Walker formulation of the LPC equations [8, 9]. Since the correlation matrix is Toeplitz (i.e. cross-diagonal symmetric), the linear system of Equation 6.6 only has $(2P - 1)$ degrees of freedom instead of P^2 . The numerically efficient method used to solve for the predictor coefficients is called the Levinson-Durbin Algorithm [8, 9]. This recursive algorithm is more efficient than matrix inversion because the linear system is Toeplitz.

6.2.2 Determining Model Order from the Spectrum

For an AR process, the predictor coefficients form a polynomial in z . This polynomial is the characteristic equation for the system resonance. Because roots of the C.E. come in complex-conjugate pairs, we need two model coefficients to estimate each spectral peak. For this reason, we choose even numbers for model order. For successful implementation of LPC, appropriate model order selection requires prior knowledge of the power spectrum. When the model order is too small, the signal is under modeled and the prediction error is not well de-correlated. The model will not have enough spectral resolution to identify distinct peaks and the energy will smear to best fit the LPC model order. When the order is too high, the matrix equation can become ill-conditioned, and the spectrum often has spurious peaks when the SNR is small.

6.2.2.1 LPC Model Order Case Study

A comparison of LPC spectral estimates of different model order (Figures 6.9-11) can be seen for a synthetic signal composed of two sinusoidal components and observed in additive white Gaussian noise. The sinusoidal components were placed at 100 and 350 Hz respectively.

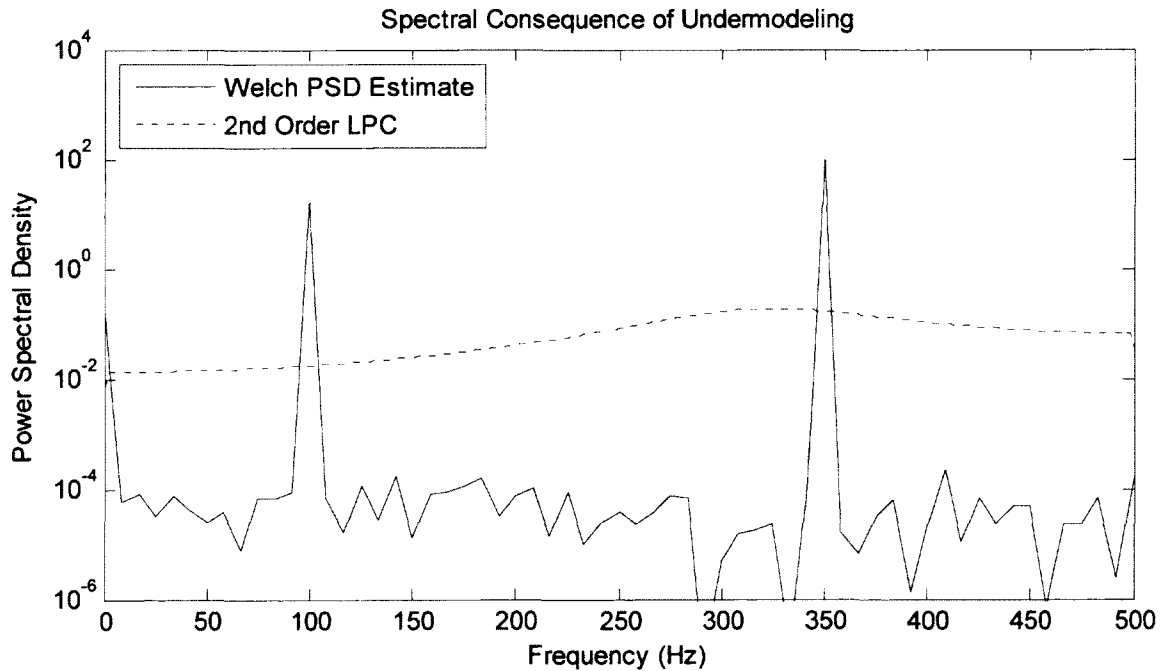


Figure 6.9 - 2nd Order LPC model smears energy across the spectrum

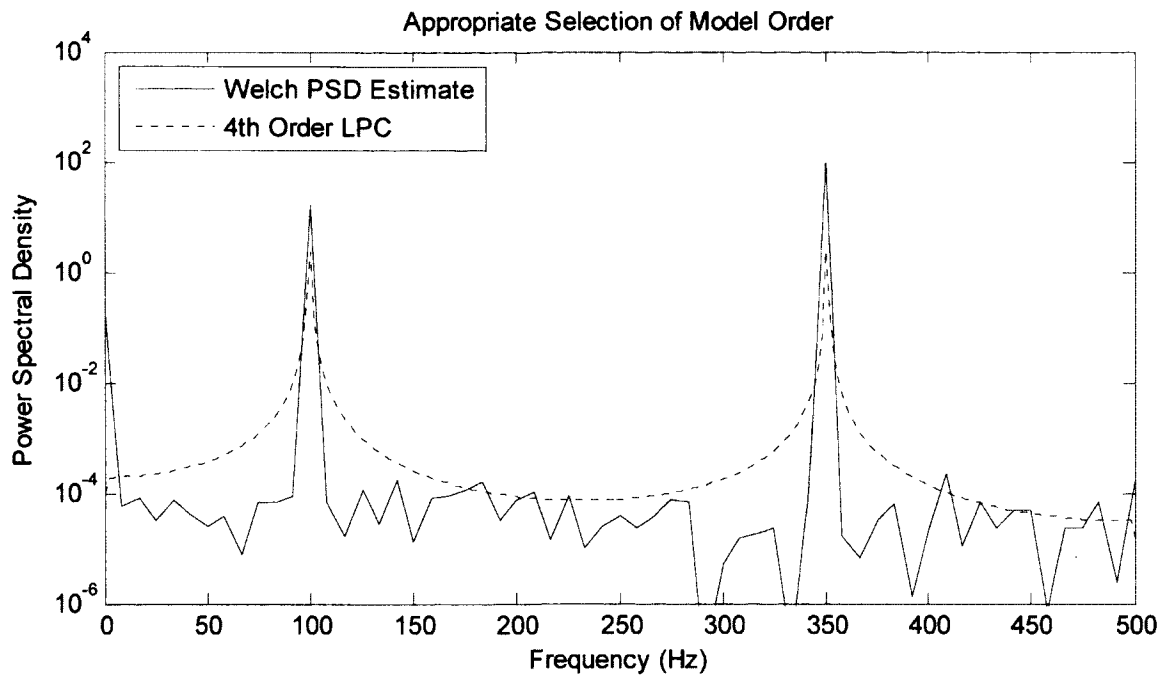


Figure 6.10 - 4th order LPC model describes resonance well

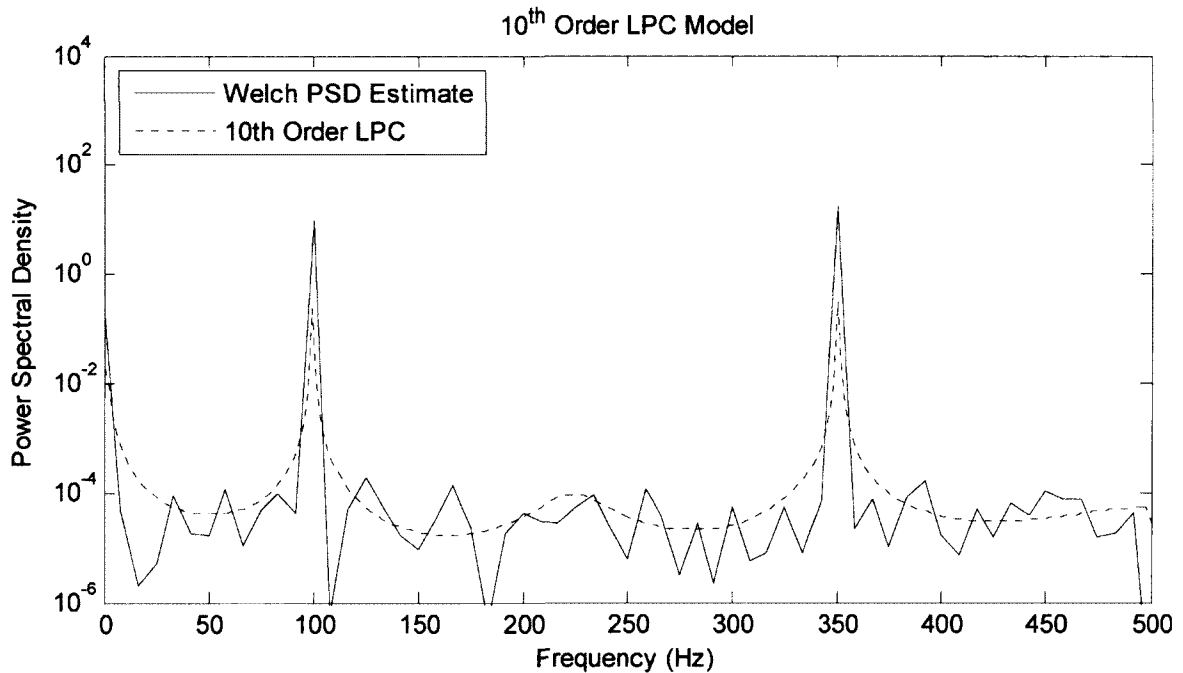


Figure 6.11 - 10th order LPC model used to describe a system with two resonant modes

We see that narrowband features are enhanced by increasing the LPC model order. If we continue to increase order, we must be cautious not to add artificial information to the estimate. In general, appropriate model order selection requires a priori information about the spectrum.

6.2.2.2 Model Order Selection for End Milling Data

Because cutting force profiles have a lot of harmonic content, the in-cut profile must be de-trended, as shown by Figure 6.12, to achieve good results from low-order LPC models. The out-of-cut signal has neither a bias nor a trend and can be used as measured to estimate open loop resonance. A 6th order model (see Figure 6.13) has been found to work well to analyze both open-loop free vibrations and closed loop vibrations from de-trended force profiles.

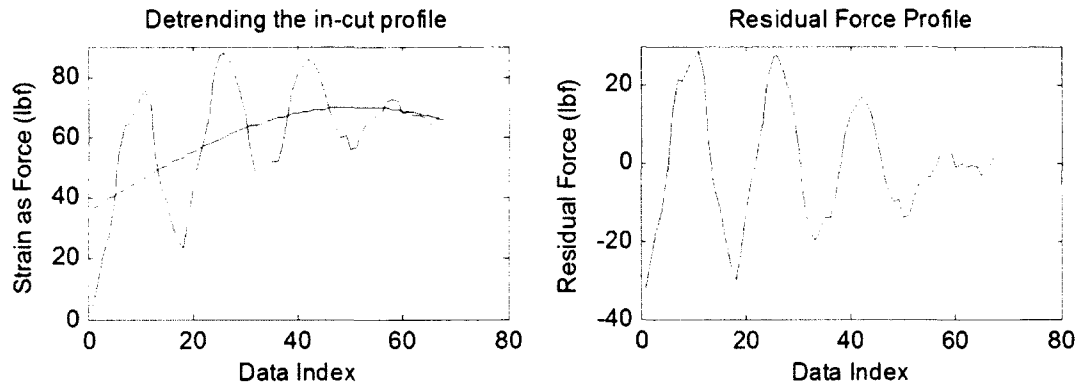


Figure 6.12 – De-trending the in-cut data for use with LPC

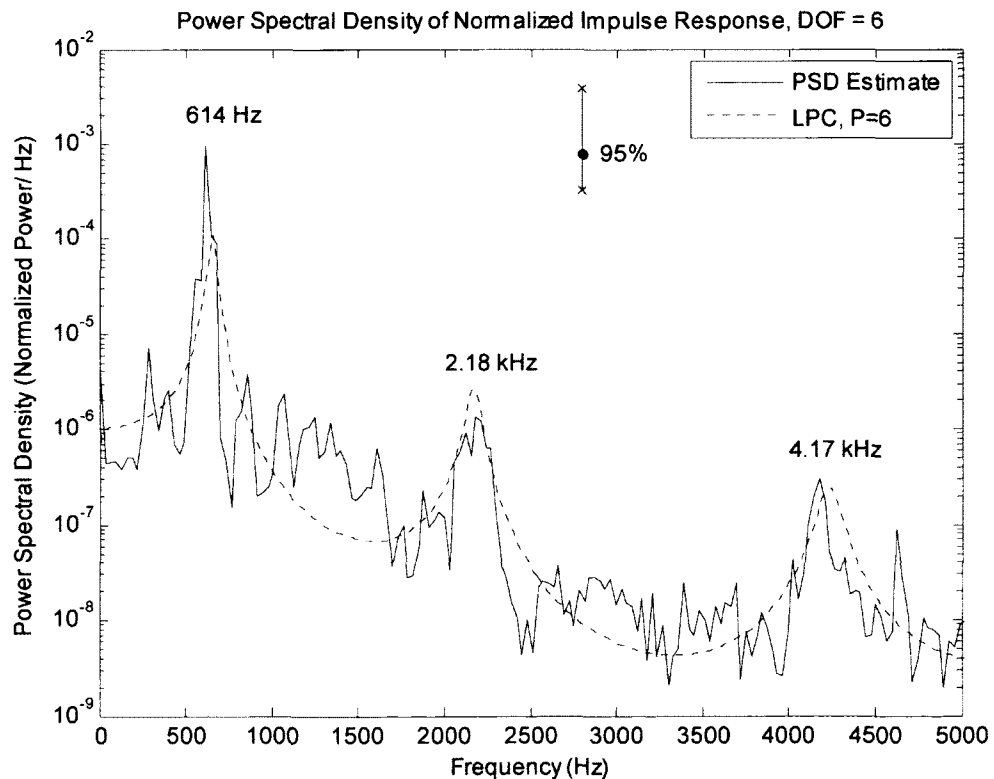


Figure 6.13 - 6th Order LPC works well for both free vibrations and de-trended force signals

As an alternate approach, LPC can be used to model the strain signal as a whole. Consider the Smart Tool's measurement signal for an example cut of three-quarter immersion upmilling at 3000 RPM with an average chip thickness of 0.002 inches (See Figure 6.14).

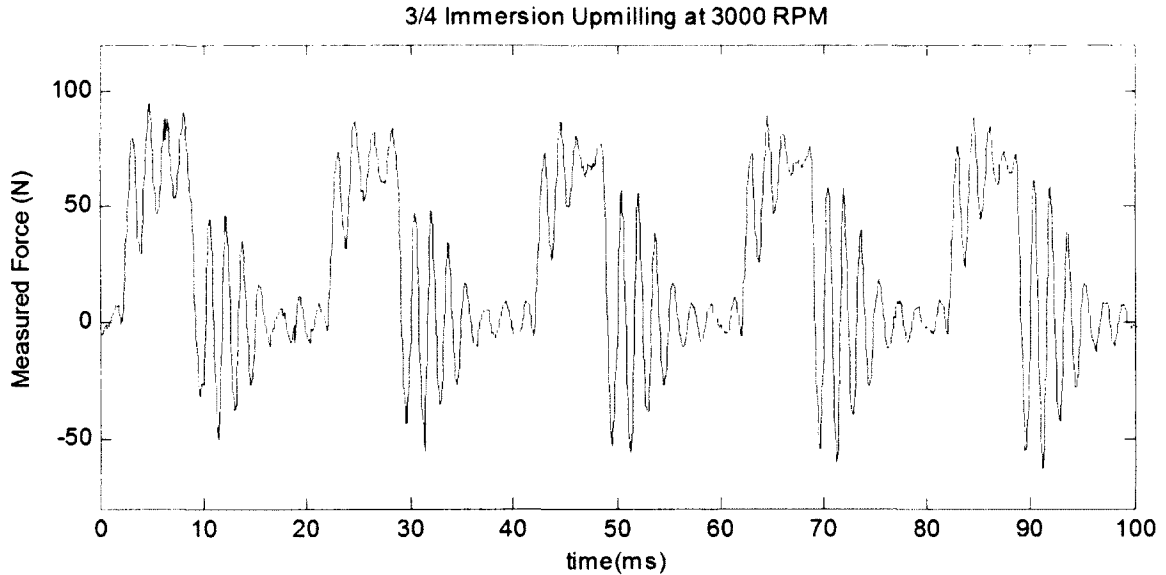


Figure 6.14 – Smart Tool's measured strain converted to force for an example cut where the force measurement has been corrupted by the system dynamics

Clearly, the shape of the measured signal is more indicative of tool deflection than it is of force. The purpose of using LPC here is attributed to its power as a tool for system identification. We can use LPC to identify an appropriate model which can subsequently be used to track resonant modes of interest, to develop linear filters that reduce unwanted vibration, or as a tool for model-based interpolation to recover sections of missing data.

Without de-trending the force profile (to remove the toothpassing harmonics), we can still use LPC to model the signal. If the damped vibrations have different ringing frequencies in-the-cut versus out-of-the-cut, this combined model will smear the energy in the spectrum to best model the combined system resonance. While a 6th order LPC model was shown above to well-model the de-trended signals, significantly larger model order is required to accurately model the measured signal because of the toothpassing harmonics. LPC estimates of increasing accuracy are shown in Figures 6.15 through 6.21.

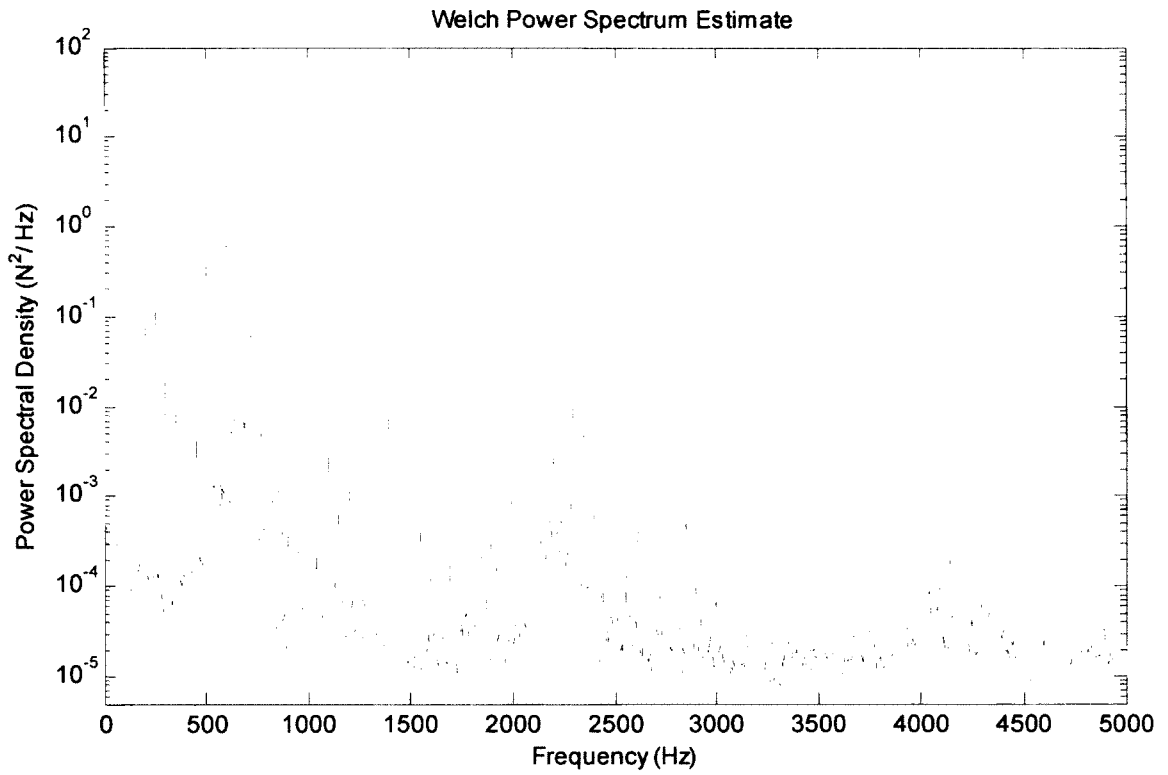


Figure 6.15 - Welch power spectral estimate of the measured force signal

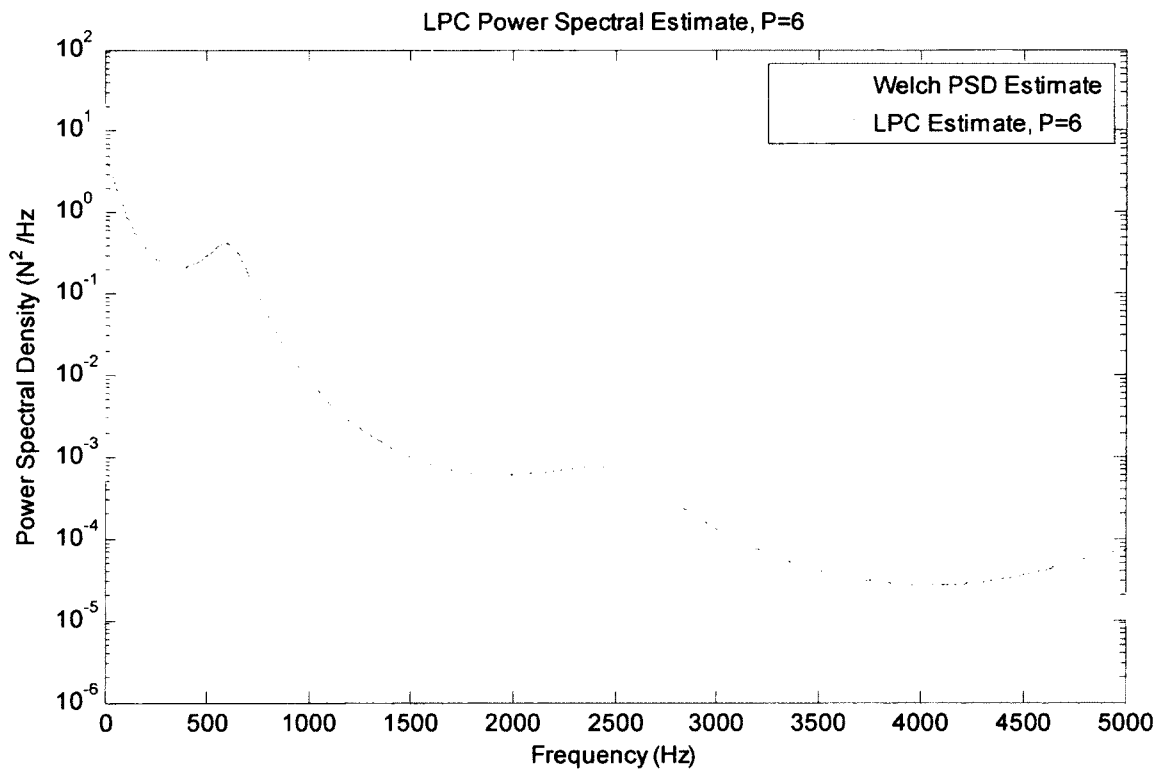


Figure 6.16 - LPC estimate of the power spectrum, P=6

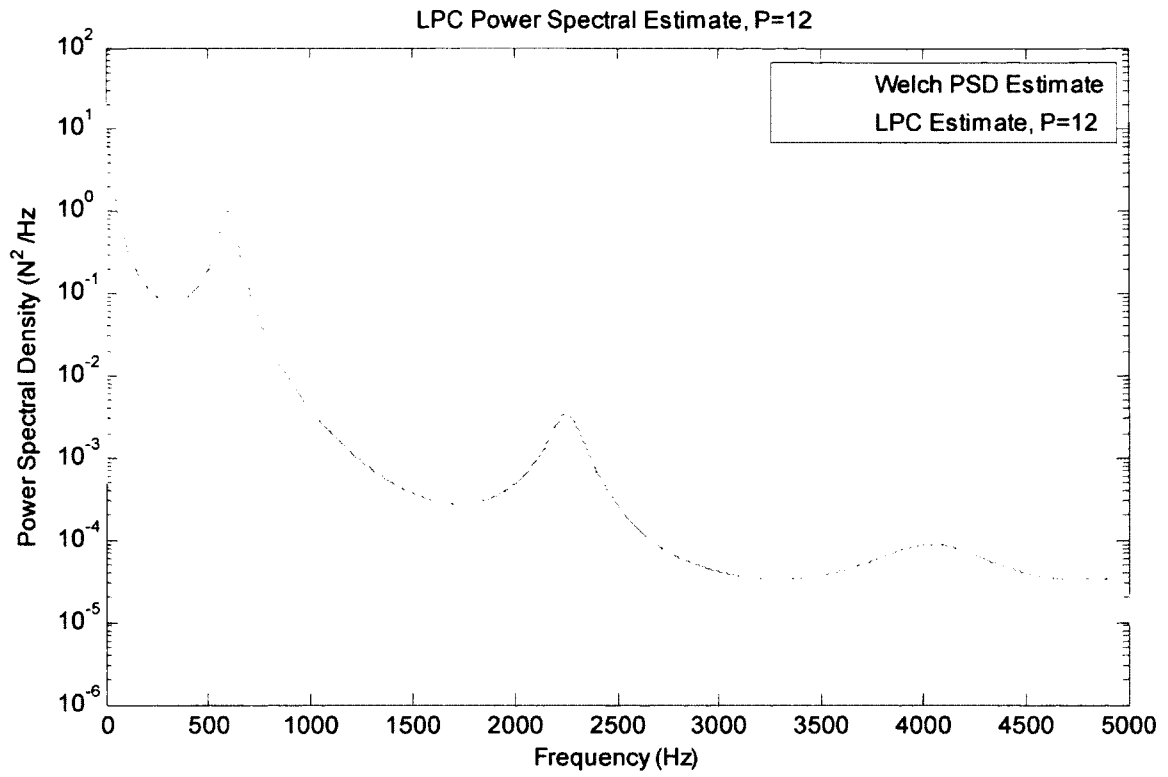


Figure 6.17 - LPC estimate of the power spectrum, $P=12$

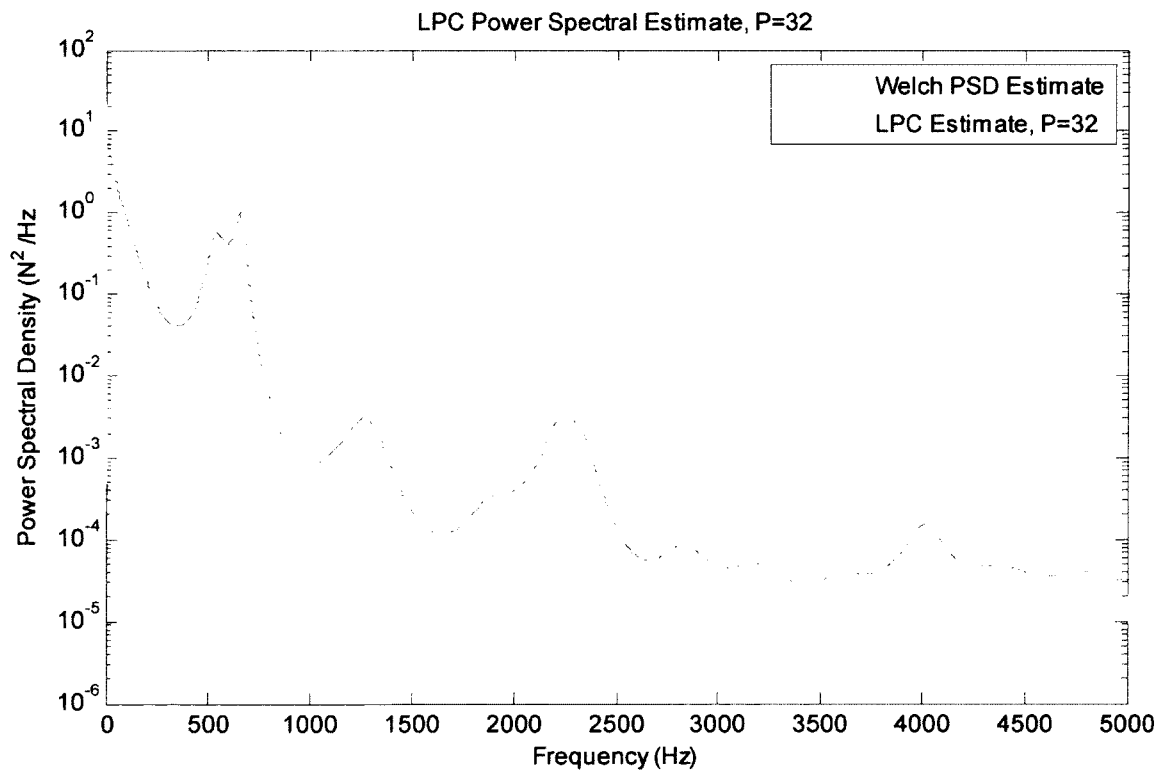


Figure 6.18 - LPC estimate of the power spectrum, $P=32$

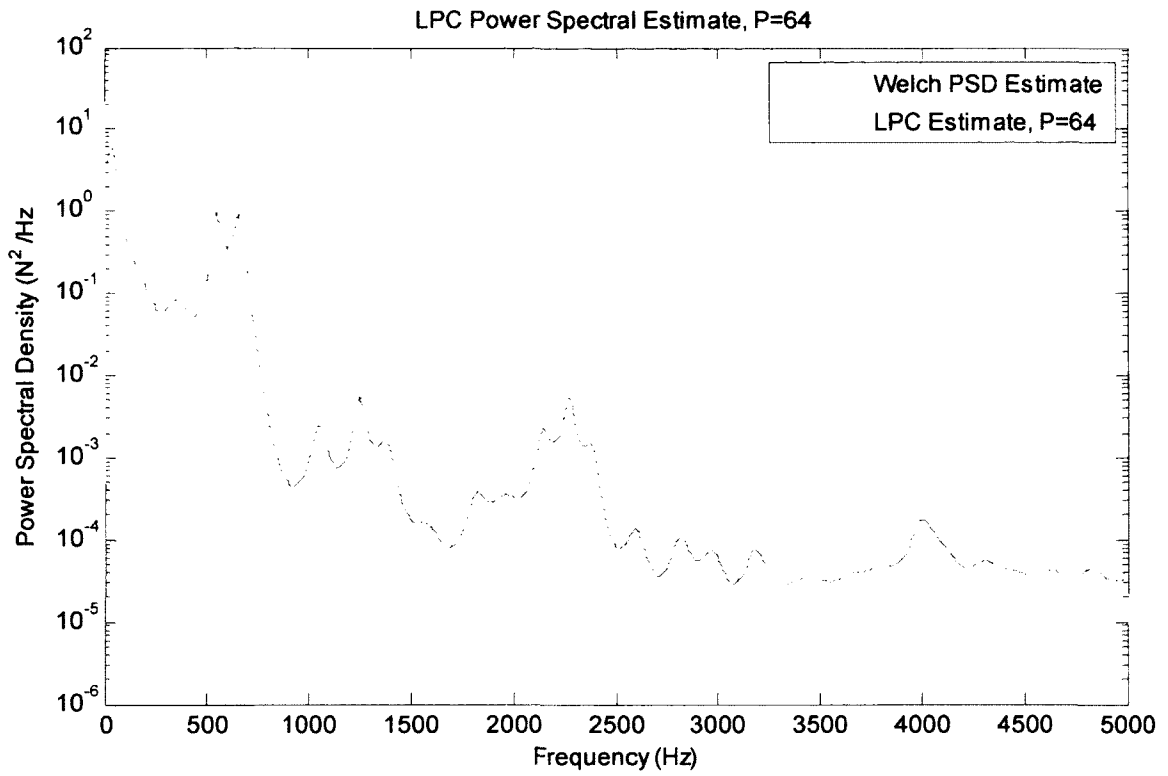


Figure 6.19 - LPC estimate of the power spectrum, $P=64$

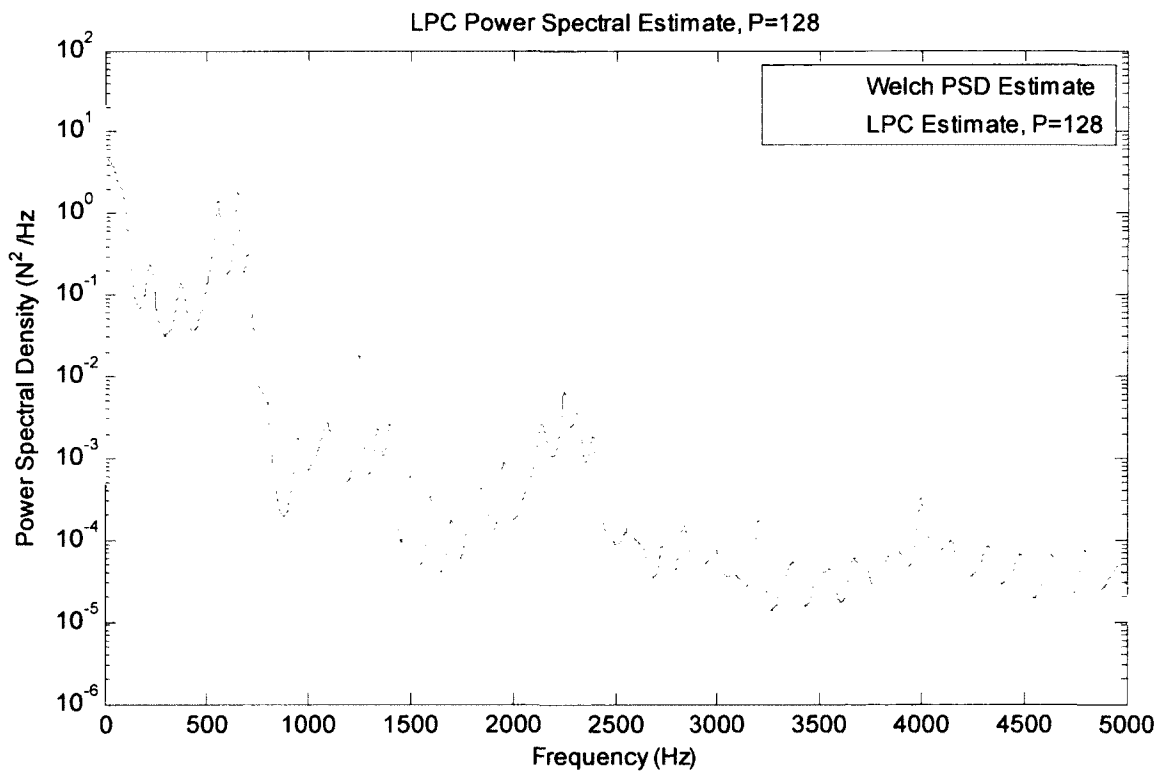


Figure 6.20 - LPC estimate of the power spectrum, $P=128$

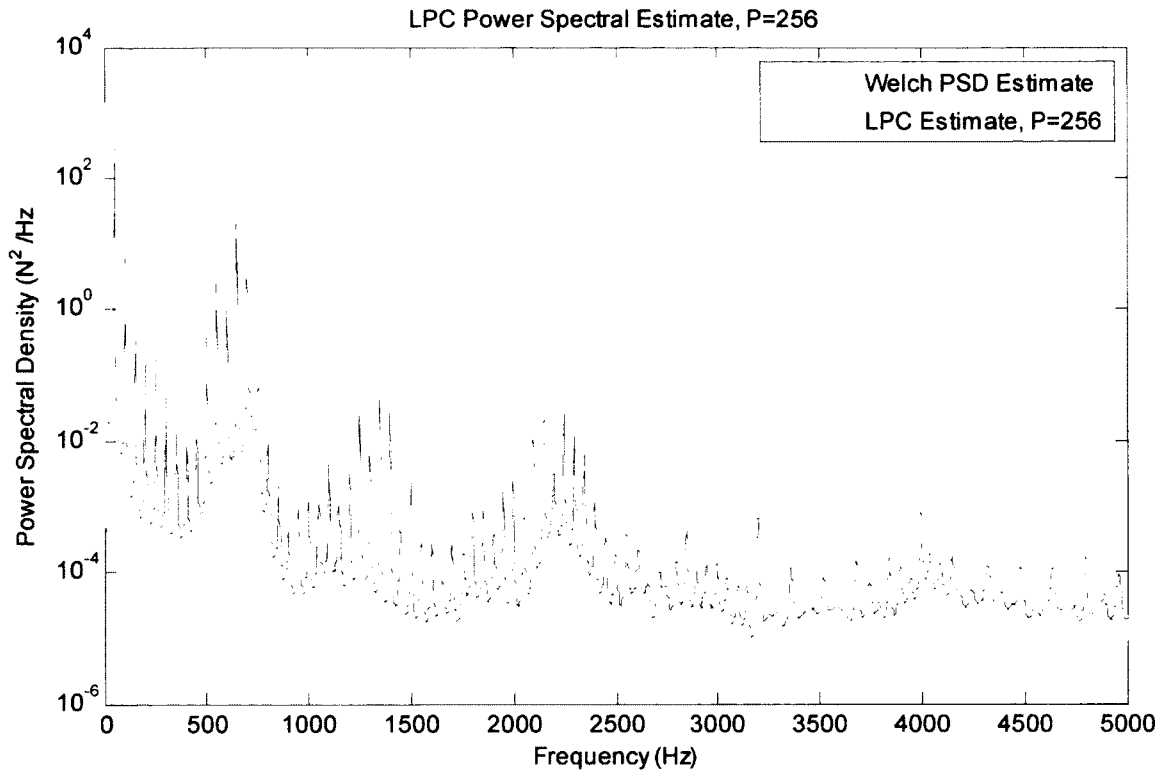


Figure 6.21 - LPC estimate of the power spectrum, P=256

To accurately model the signal as a whole, the LPC model order should be larger than the number of samples per revolution. The usefulness of modeling the signal as a whole (as opposed to using in-cut and out-of-cut section independently) is that a complete model of the cutting signal can be used to perform Least-Squares Auto-Regressive (LSAR) interpolation [9] if any of the strain data is lost or corrupted during transmission.

It may also be of interest to implement a low-order model on the signal (like P=12 in Figure 6.17) to track resonant modes of vibration if it is computationally too-expensive to break the signal into in-cut and out-of-cut sections for more accurate system identification.

6.2.3 Case Study: Dynamic Parameter Estimation for $3/4$ Immersion Upmilling at 3000 RPM

As stated earlier, dynamic parameter estimation is needed to track the system dynamics; with a model of how the system is resonating, we can successfully implement model-based filtering techniques. The following presents how Linear Predictive Coding can be employed to track the open-loop (out-of-cut) and closed-loop (in-cut) system dynamics from the Smart Tool's bending strain signal. The general approach is illustrated by Figure 6.22.

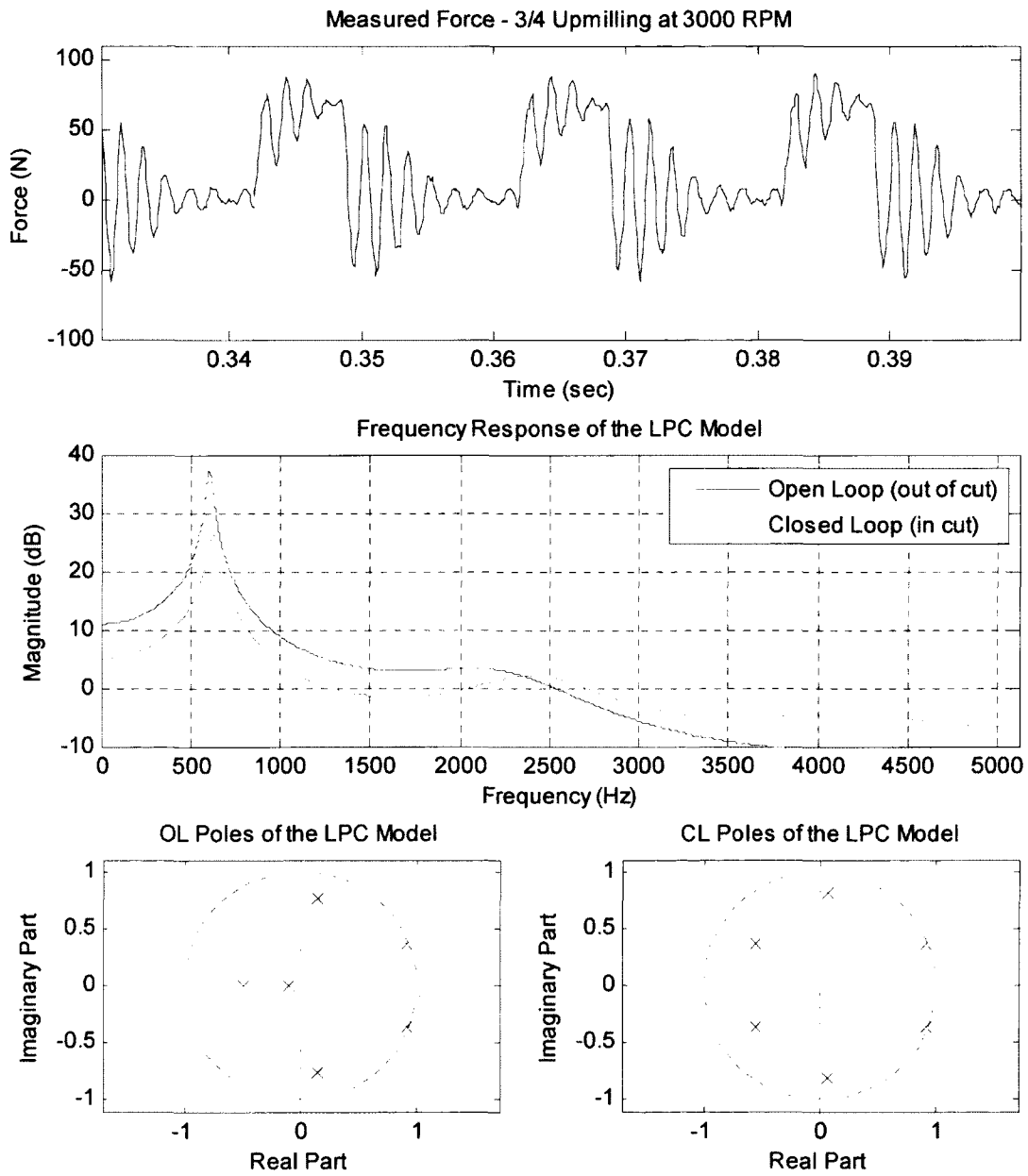


Figure 6.22 – LPC used to implement auto-regressive modeling of dynamic system resonance

Figure 6.22 shows the system dynamics estimated by the LPC algorithm for a single revolution of the tool. The in-cut profile is de-trended as shown by Figure 6.12, and LPC is used on the in-cut profile and the out-of-cut free vibration separately. The bottom plots show the estimated pole locations of the open-loop and closed-loop AR models, and the middle plot shows the frequency response of these LPC models.

The results of this parameter identification agree with our intuition: The fundamental mode of vibration has more damping in the cut, and the corresponding ringing frequency is slightly higher than the open loop mode because the boundary condition changes when the tool engages with the workpiece. This agrees with the notion that the dynamic stiffness increases when the tool is engaged with the workpiece. Implementing the LPC algorithm for every tool rotation allows us to track how the dynamics vary with time, as seen in Figure 6.23.

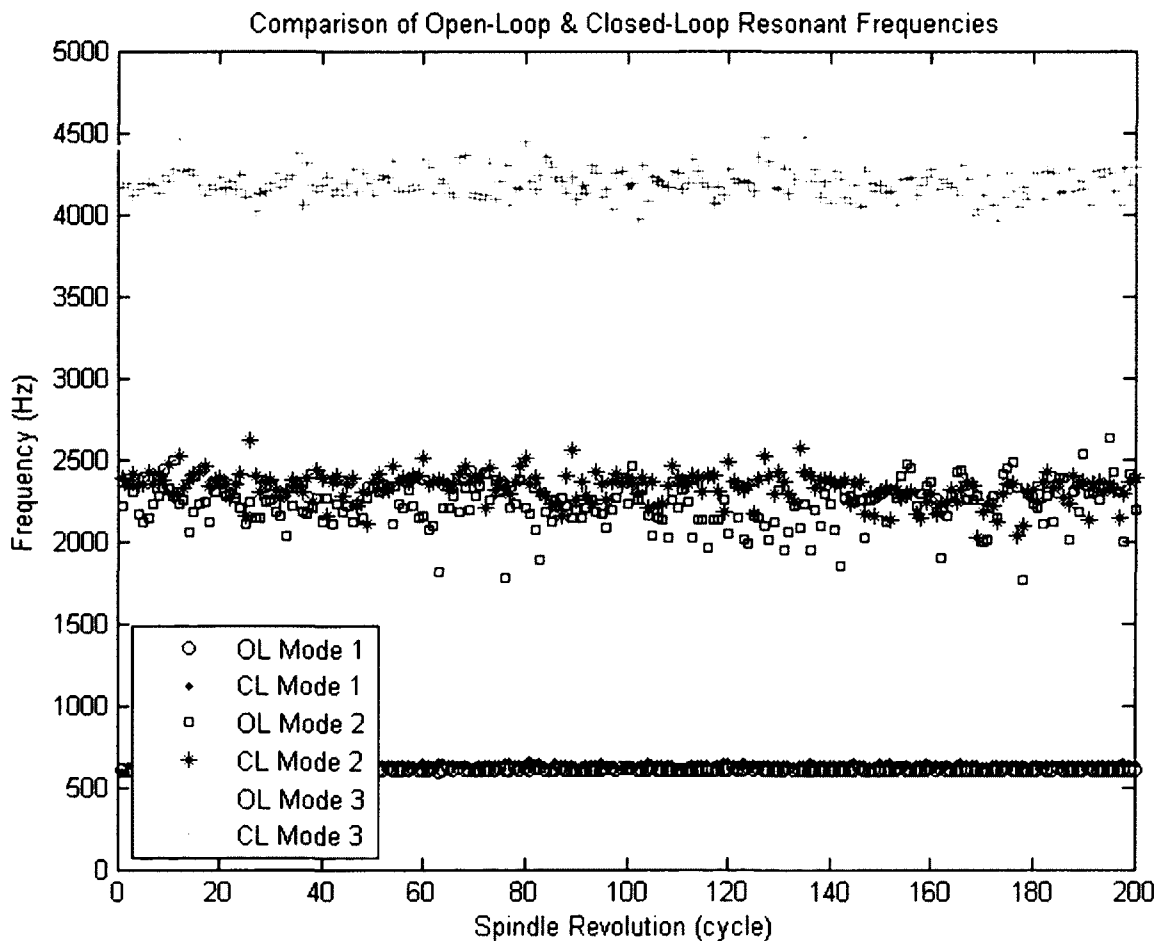


Figure 6.23 - Comparison of open-loop and closed-loop resonant frequencies

Figure 6.23 tracks the resonant modes of the in-cut and out-of-cut data over two seconds (200 cycles with 50 Hz toothpassing frequency). As shown in Figure 6.22, most of the energy in the signal is contained in the fundamental mode which remains relatively constant compared to the higher modes, but still changes too much to be removed with a time-invariant filter. A close-up view of how the fundamental mode changes in time is shown below in Figure 6.24.

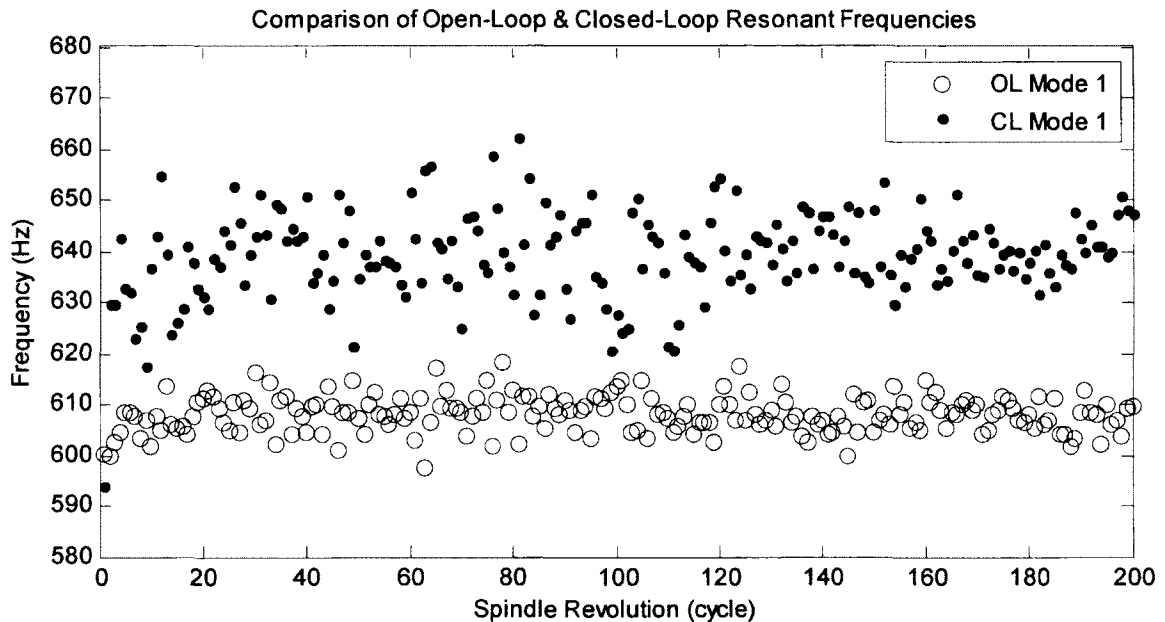


Figure 6.24 - Tracking variation in the fundamental resonant mode of tool vibration

Intuition suggests that the variation in the damped natural frequency from cycle to cycle is truly as stochastic as shown above. Future work should investigate the variance of LPC estimates for a stationary signal to validate this claim.

6.2.4 Summary of LPC Technique for Dynamic System Identification

Linear Predictive Coding is a powerful tool for parameter estimation of systems with resonant spectra, because they are well described by an AR model. This technique gives us the power to inform an adaptive model-based filter about the system resonance cycle by cycle. No work has been done to investigate the potential for implementing this technique in real time. Notwithstanding, we have a robust tool for system identification that makes the Smart Tool useful for research applications where the data only needs to be interpreted during post-processing.

6.3 A Simple Algorithm for Enhanced Chatter Frequency Detection

Fourier-based spectral analysis is the most common technique used to identify modes of vibration in end milling. One of the problems associated with looking at the spectral content of the measured force, however, is that the harmonics of the toothpassing frequency degrade our ability to interpret the spectrum and identify frequencies corresponding to forced vibrations, and possibly, the onset of chatter. While it is possible to minimize these unwanted harmonics with a comb filter, it would be nice if we could somehow gain insight into the system dynamics without the need to filter the signal first.

A simple way to remove these unwanted harmonics entirely is to difference the measurement signal by subtracting the instantaneous measurement from the previous revolution. This approach is illustrated below in Figure 6.25.

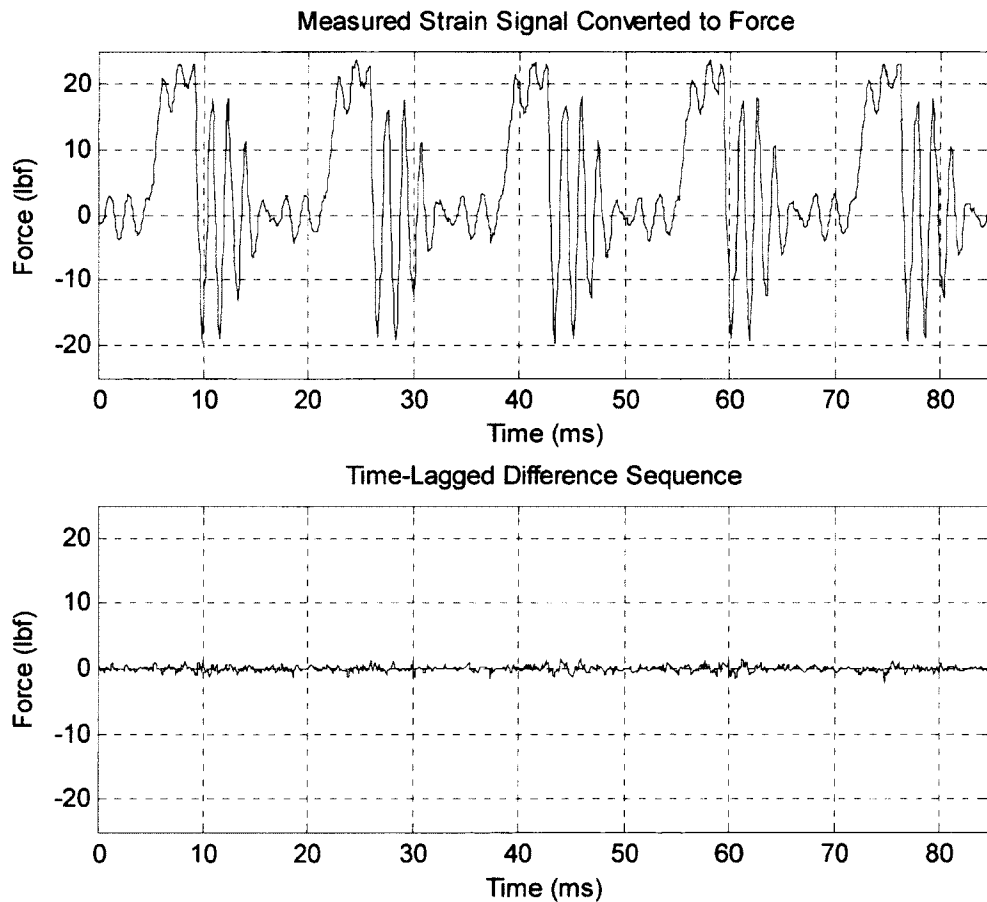


Figure 6.25 - Time-lagged difference sequence represents the dynamic chip thickness

The challenge here is that the sampling frequency is often not a perfect multiple of the spindle speed; we must know the spindle speed exactly (or estimate it from the data), and interpolation is required to obtain the value of the measurement from the previous revolution. The result, however, is that we entirely remove the harmonics associated with the toothpassing frequency, and spectral analysis allows for enhanced chatter frequency detection.

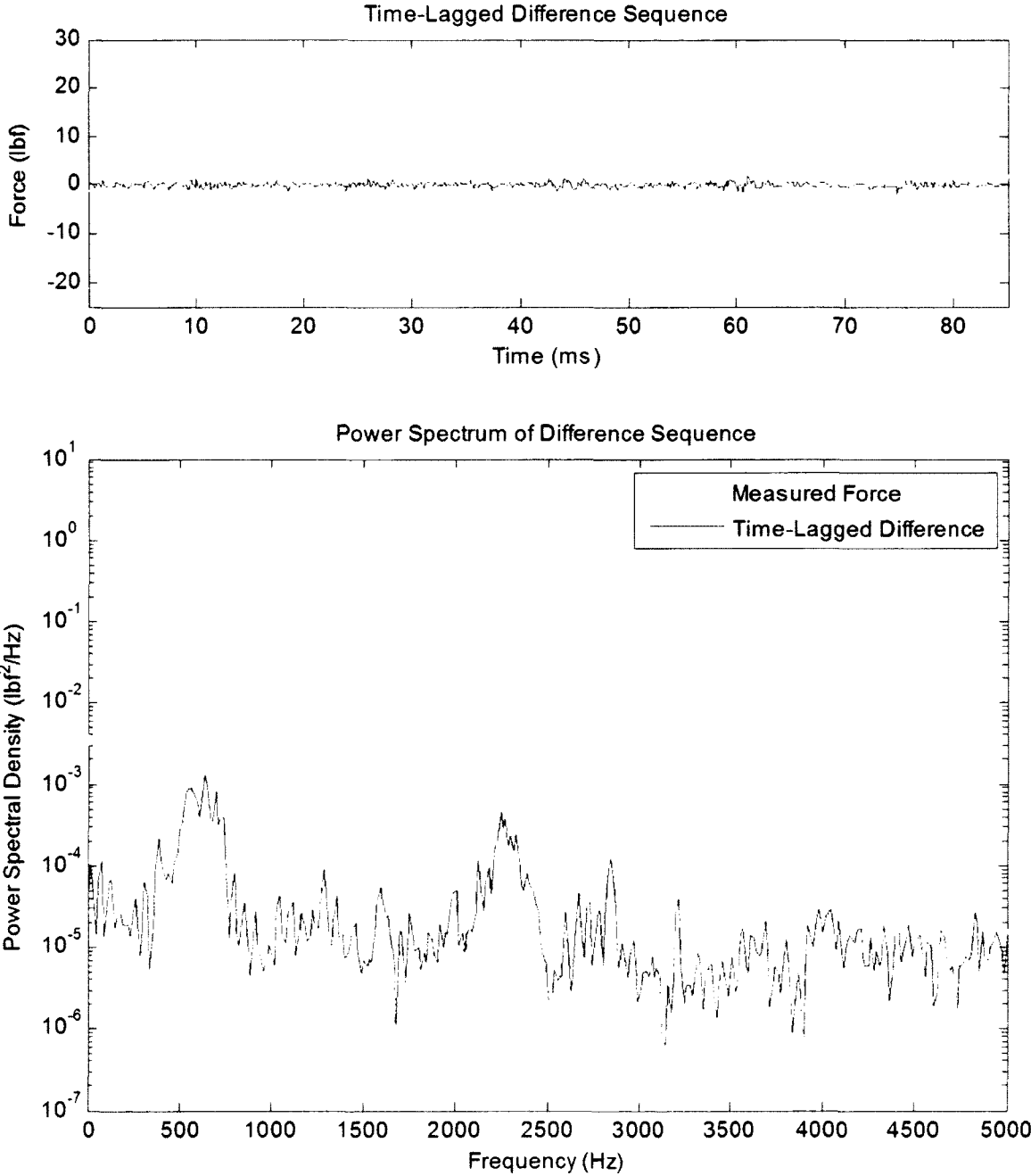


Figure 6.26 - Enhanced chatter frequency detection is achieved by differencing the measurement signal

Figure 6.26 shows that the toothpassing harmonics (which interfere with our ability to locate possible chatter frequencies) are almost entirely removed by looking at the spectrum of this “time-lagged difference sequence.” This method is superior to that of using a comb filter, because the signal is conditioned in the time domain. There is no need to worry about spectral leakage, smoothing, appropriate data windows, etc, because the unwanted periodic components are completely removed. Now, comparing the spectra of the measured force and the difference sequence, we can clearly locate the spectral peaks in the difference sequence corresponding to modes of system resonance. These peaks are less evident in the spectrum of the measured force.

An even simpler technique is to use a first-order difference to remove the force profile,

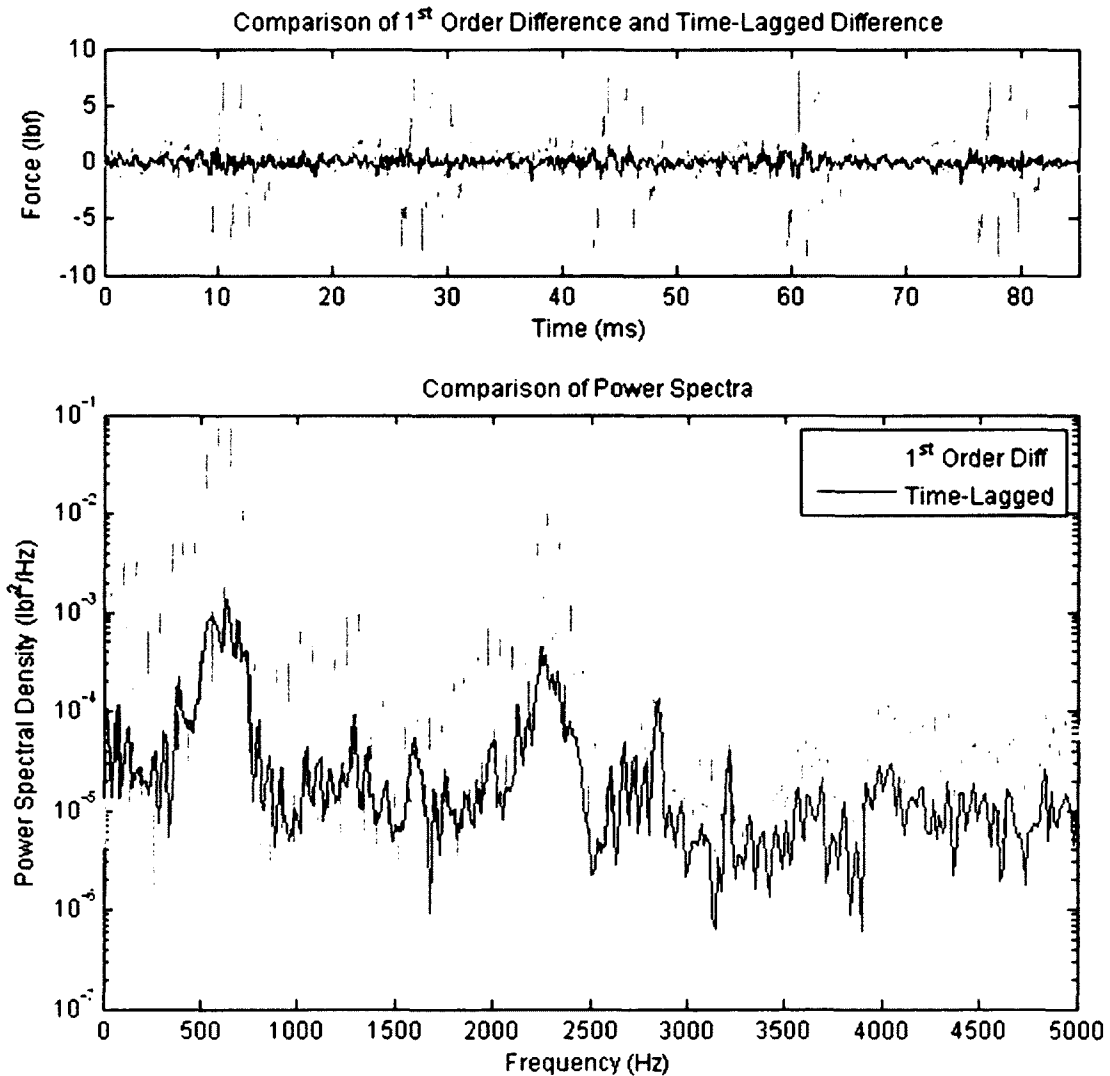


Figure 6.27 - Comparison of differencing estimators

as seen in Figure 6.27. A first-order difference is formed by taking the difference between two consecutive data points. As seen in Figure 6.27, while the first-order difference appears to do a good job of removing the force profile in the time domain, the spectrum still contains artifacts of the toothpassing frequency and its harmonics. Higher-order differences were implemented as well, but they showed no improvement in the spectrum.

This time-lagged differencing algorithm is a powerful technique because it completely removes the periodic components of the force profile. Since the bending strain signal is mostly characteristic of tool deflection, the result of subtraction from one revolution prior creates a signal that is largely indicative of the shape of the dynamic chip load (in the cut). With knowledge of the spindle speed, it is a simple matter to difference the measured data – the result is the ability to perform enhanced chatter detection by completely removing the interfering harmonics.

6.4 Model-Based Optimal Filtering

This section considers optimal techniques in signal processing that can be used to artificially extend the bandwidth of the sensor. The techniques presented include the harmonic Kalman Filter and Weiner equalization. These optimal linear filters use least-squares techniques to combine the best of system modeling and actual measurements. Of course, the performance of any “optimal” technique is only as good as the underlying model.

The techniques presented here are considered optimal because the filter structures are derived by minimizing some sort of Bayesian risk function (i.e. cost function) [11]. For example, minimizing the mean squared error gives the Bayesian MAP (*Maximum A Posteriori*) estimate corresponding to the conditional mean of the estimated parameter computed using the posterior probability density function; minimizing the absolute value of error gives the Bayesian MAVE (*Minimum Absolute Value of Error*) estimate corresponding to the median of the posterior probability density function. Note that if the noise is Gaussian, these estimators are identical.

6.4.1 An Introduction to State Estimation

The general problem we now face is estimation of a signal that has been distorted by some dynamical system and observed in additive noise (Figure 6.28).

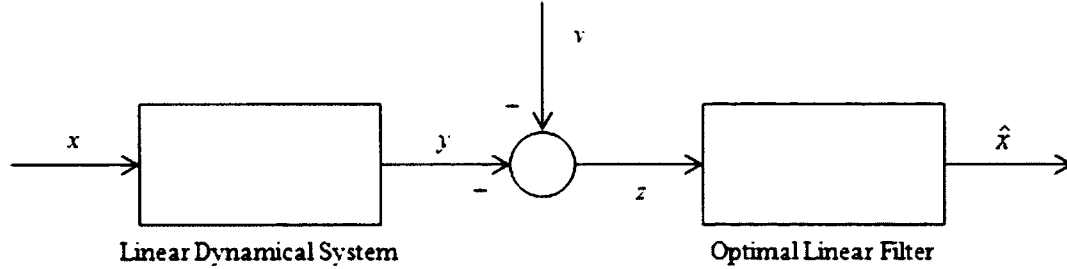


Figure 6.28 - Problem formulation for state estimation

Here, x is the true process we wish to know, y is the process x after being transformed by the dynamic system, v is additive white Gaussian noise, z is the noisy measurement, and \hat{x} is our estimate of the true process x . For optimal estimation, a quadratic cost function of the state residual would be the most meaningful,

$$J(x) = \frac{1}{2} \boldsymbol{\varepsilon}_x^T \boldsymbol{\varepsilon}_x = \frac{1}{2} (x - \hat{x})^T (x - \hat{x}) \quad (6.7)$$

but the reason for estimation is that x is unknown. Hence, x is normally unavailable for cost function evaluation, and for formulation of the estimation equations. On the other hand, the measurements are available, and with the reasonable assumption that z bears some systematic relationship to x ,

$$\begin{aligned} J(z) &= \frac{1}{2} \boldsymbol{\varepsilon}_z^T \boldsymbol{\varepsilon}_z = \frac{1}{2} (z - \hat{y})^T (z - \hat{y}) \\ &= \frac{1}{2} (z - H\hat{x})^T (z - H\hat{x}) \end{aligned} \quad (6.8)$$

is a useful cost function. Here, H is the observation matrix of the state model. It can be evaluated without prior knowledge of x , and it can be minimized to derive \hat{x} from z . If the mean value of the noise is zero, then $J(z)$ minimizes $J(x)$ in the limit that the number of observations goes to infinity [11].

6.4.2 Harmonic Kalman Filter

The Kalman Filter is a recursive optimal filter that propagates the conditional probability density function from one sampling instant to the next, taking into account system dynamics and inputs, and incorporates both measurements and measurement error statistics in the estimate. Computing the weighting factors (or filter gains) that optimally combine measurements and extrapolations is a crucial intermediate step in the computation. The estimate is obtained by taking the mean (i.e. expected value) of the conditional density function, and the covariance matrix is used to specify the spread (or uncertainty) in the estimate [11]. The recursive formulation of the mean and covariance can be expressed in five equations:

1. State Estimate Extrapolation (Propagation)
2. Covariance Estimate Extrapolation (Propagation)
3. Filter Gain Computation
4. State estimate Update
5. Covariance Estimate Update

As Stengel [11] explains, “Given the state estimate from a previous iteration, (1) uses the dynamic process model to propagate the estimate of the state mean value to the next sampling instant without regard to new measurements. (2) does the same thing for the state covariance matrix, assuming that the “process noise” of known covariance is forcing the system. The result of (2) enters the computation of the optimal filter gains. The filter gain computation (3) weights prior knowledge of measurement error covariance with state estimate covariance on a purely statistical basis. The actual measurements have no effect on the gain computation. These measurements correct the state estimate in (4), adding the product of the gain matrix and the measurement residual to the state estimate propagated by (1). A similar correction is made to the covariance estimate (5), accounting for the known covariance of measurement errors.”

Block diagrams showing the appropriate computations for Kalman state estimation and filter gain computation are shown in Figure 6.29 and Figure 6.30:

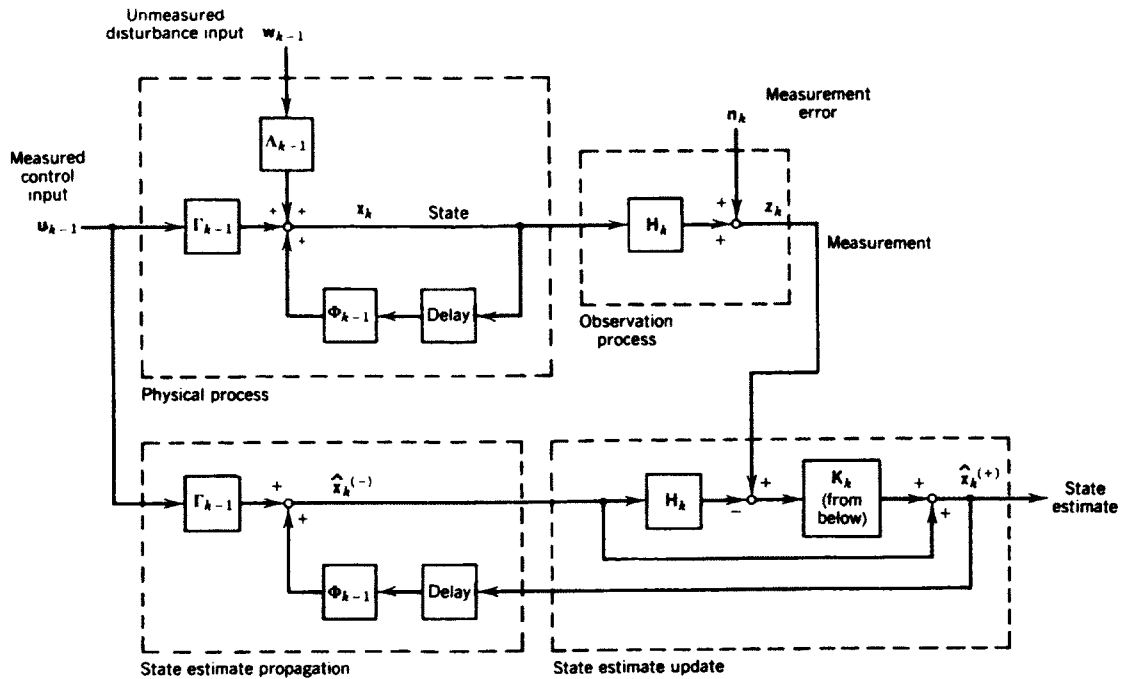


Figure 6.29 - Discrete-time system and linear optimal filter. (a) Dynamic system and state estimator [11]

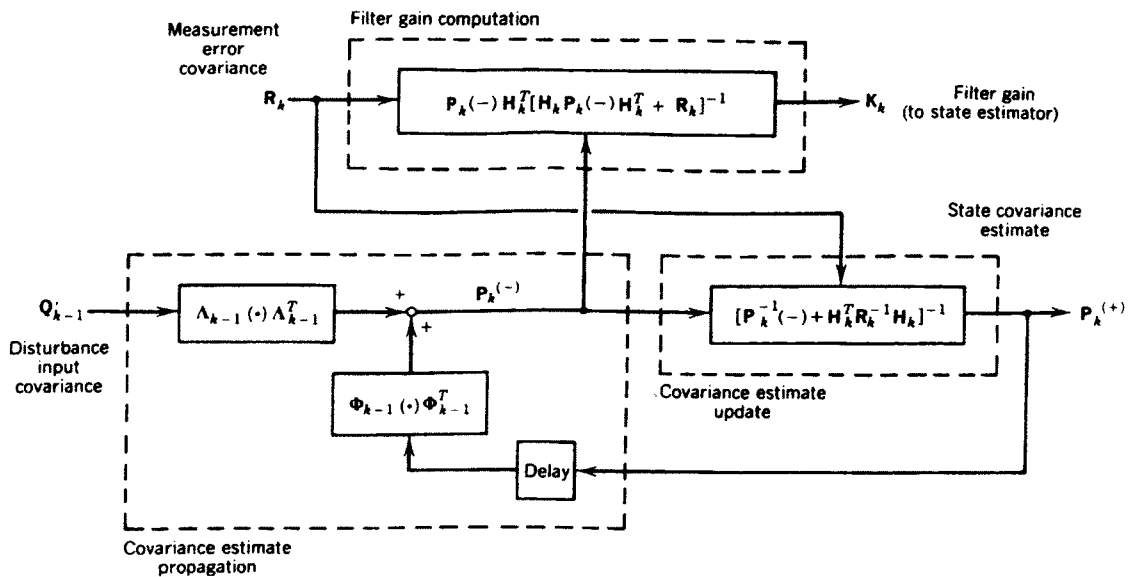


Figure 6.30 - Discrete time system and linear-optimal filter. (b) Covariance estimator and gain computation [11]

Implementation of the discrete Kalman Filter is simplified for steady-state conditions when the statistics for the driving noise and measurement noise are stationary. In this case, the

filter gains can be pre-computed, and filter implementation only requires the recursive loop shown in Figure 6.29 (state propagation and optimal state correction based on filter gains).

6.4.2.1 The Linear Dynamical Model

Our motivation for using the Kalman Filter is to artificially increase the bandwidth of the sensor. Recall that we want to do this so that we can separate the measurement into a static component useful for calibration, and a compliant component useful for monitoring the system dynamics. This is readily accomplished by building a system model that looks like the cutting force resulting from the static chip thickness (Figure 6.31).

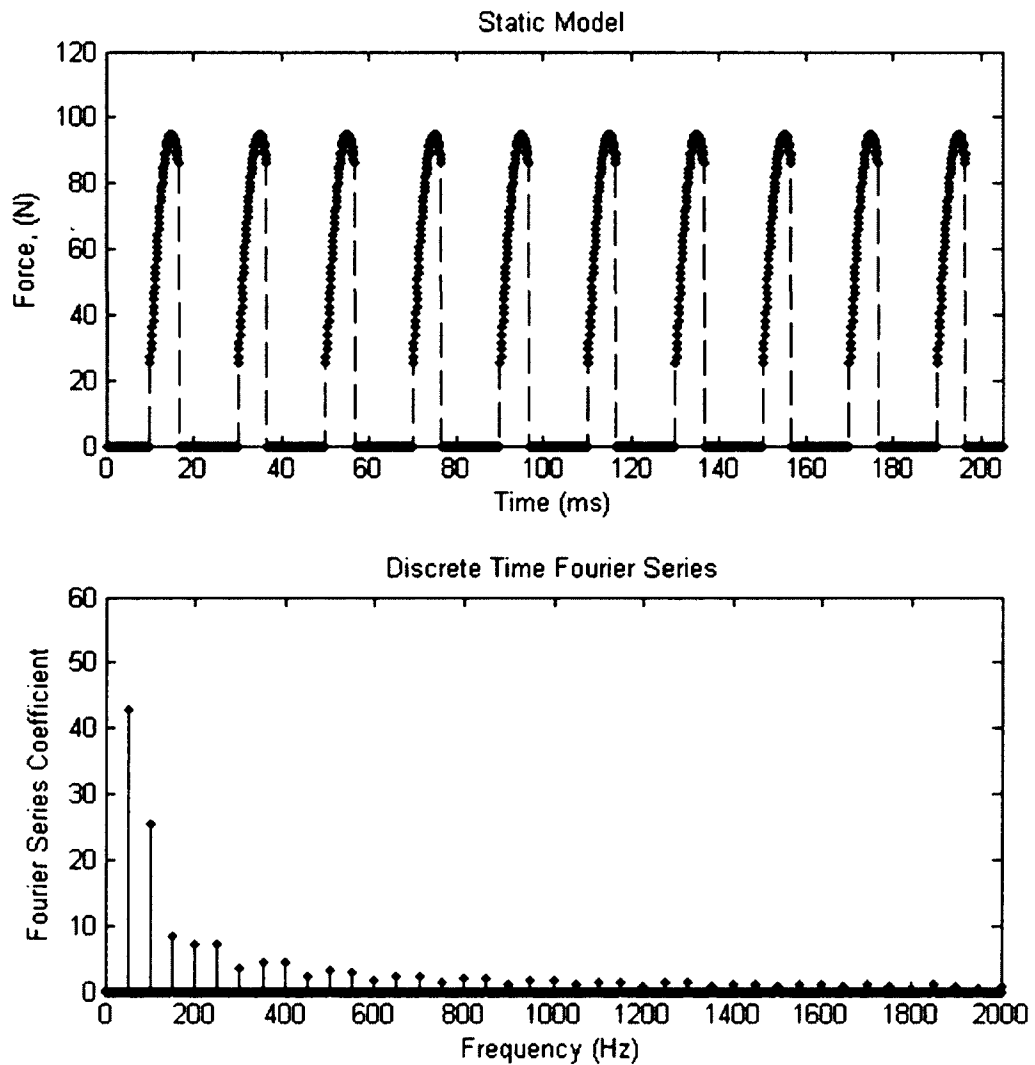


Figure 6.31 - Discrete Fourier Series representation of the static force profile

We implement our model of the “static force” profile by using a series of oscillators corresponding to the toothpassing frequency and its harmonics. We can exploit our knowledge of the spectrum during model building to make our lives easier: Since our objective is to build a state model with the appropriate frequencies and magnitudes to describe these harmonics, we can simulate the static profile such that the synthetic signal has an integer number of points per cycle, and an integer number of complete cycles. When we do this, the spectrum computed using the FFT is exactly the Discrete Fourier Series (DFS) [9]. To this end, we have a *perfect* model in the frequency domain of the discrete sinusoids needed to recreate the static force profile.

There is an inherent tradeoff between accuracy and model complexity using this technique. As frequency increases, the spectral contribution to the total energy of the signal is also seen to decrease. We must make a design choice as to how many spectral components we will use to build the linear model. Our decision must consider the fact that the estimation algorithm will only work as well as the underlying model. This design choice is aided by looking at the integral of the power spectrum of the static force model (Figure 6.32).

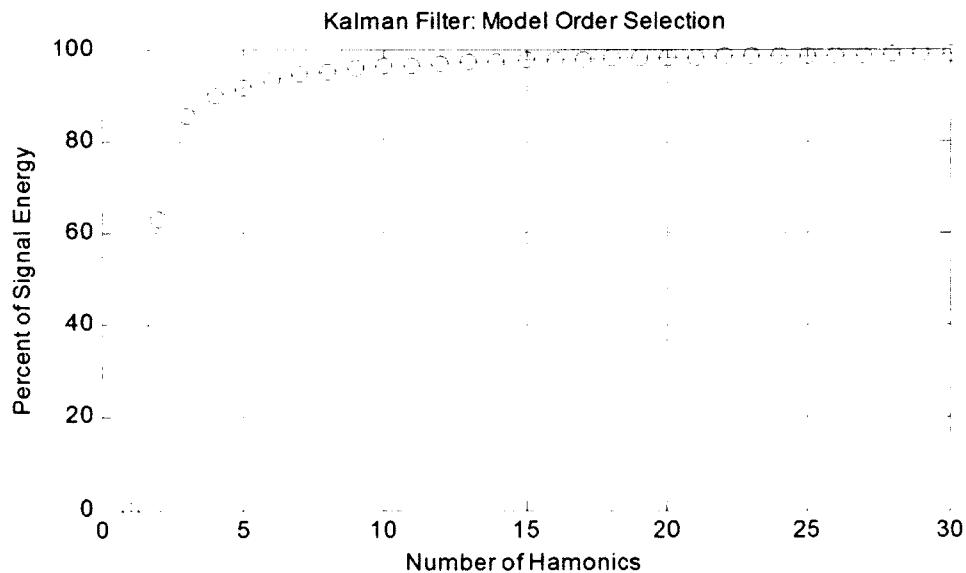


Figure 6.32 - Numerical integration of the static model spectrum is used to specify model order

We see that a model with 10 harmonics captures 96% of the energy contained in the static signal. Increasing model order beyond this point does not add much information to the model.

6.4.2.2 Filter Gain Computation

The filter gains are computed solely from the statistics of the driving “process noise” and the additive observation noise. These gains are computed to minimize the mean square error between the model and the observations assuming that the measurement noise is Additive White Gaussian Noise (AWGN). This is most readily accomplished using the MATLAB built-in function “Kalman.m”. Complete details of the mathematics can be found in [9, 11, 12, 13].

6.4.2.3 Filter Implementation

With a model of the linear system and knowledge of the noise statistics, we can build this “Harmonic Kalman Filter” and tune the gains to achieve the desired frequency response. The Bode plot of an example filter tuned for radial force measurement during 3000 rpm $\frac{3}{4}$ immersion upmilling with an average chip thickness of 0.002 inches is shown below in Figure 6.33.

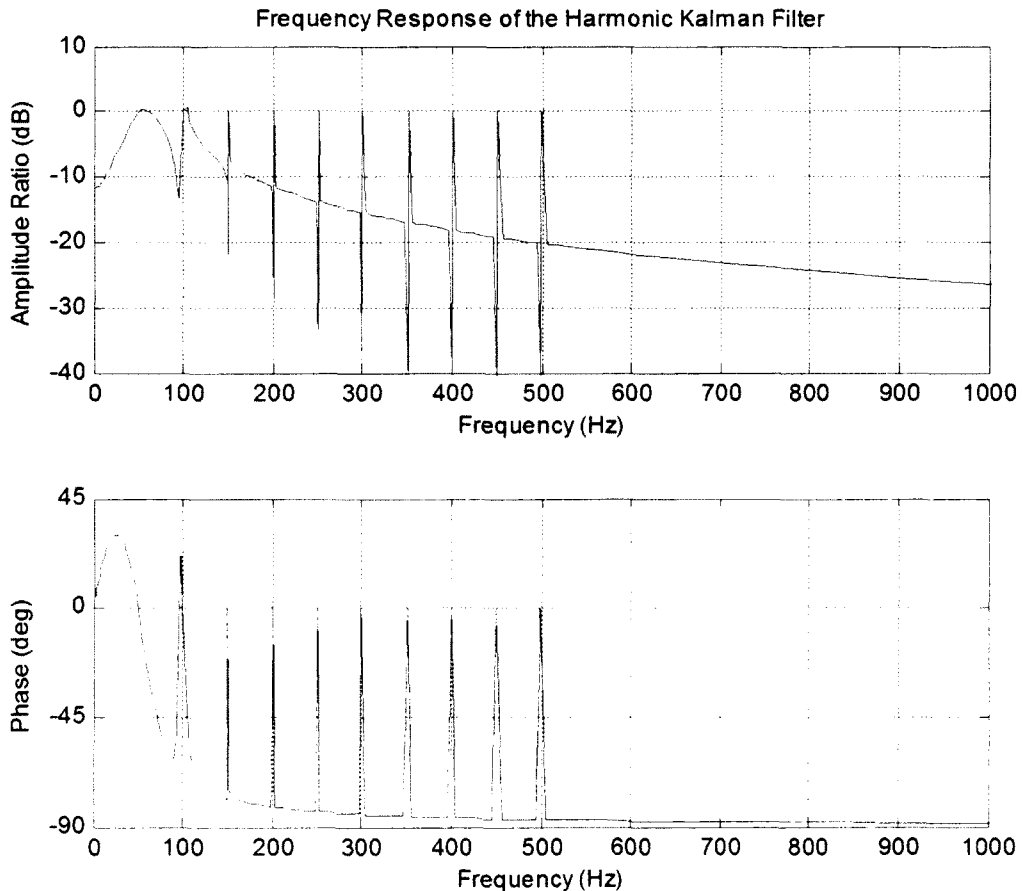


Figure 6.33 - Frequency response of the Harmonic Kalman Filter

The result of filtering the Smart Tool's measured strain signal (3/4 immersion upmilling at 3000 RPM, radial force) with our harmonic Kalman Filter is shown below in Figure 6.34.

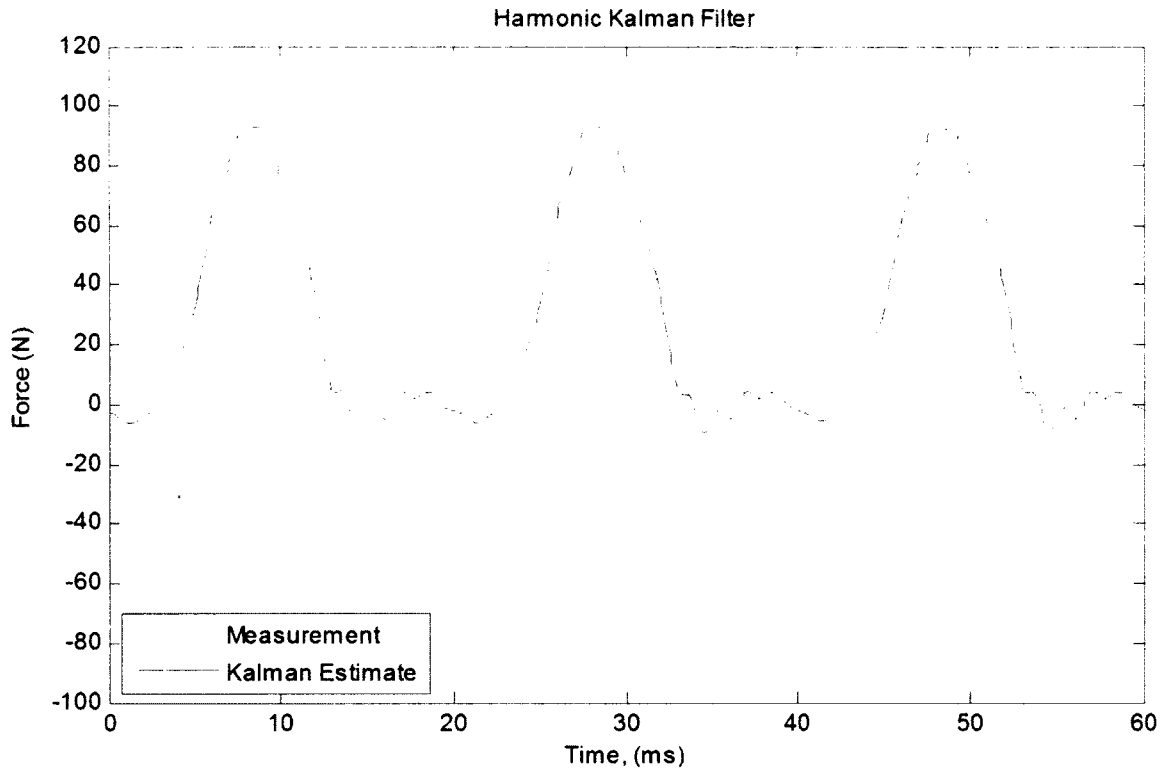


Figure 6.34 - Implementation of the Harmonic Kalman Filter

We first observe that our Kalman filter does a reasonable job of artificially increasing the system bandwidth. Vibrations are significantly attenuated, and peak force estimates are maintained; this is the general advantage of the Kalman filtering technique versus a simple low pass filter – the peak of the filtered signal matches that of the measurement (averaging vibrations), and there is no time delay to the signal. There is no delay because the Kalman filter is able to achieve zero phase in the pass band.

The drawback of this technique, however, is that while 10 harmonics capture 96 percent of the *energy* in the signal, all of the un-modeled harmonics are necessary to replicate the sharp corners seen in the static profile. This 10-harmonic model smears out some of this energy, widening the base of the signal and missing the high frequency content at the peak of the profile.

This smearing is unavoidable, however, because significantly increasing model order causes problems with numerical stability as the state matrix (A matrix) becomes close to singular. Nonetheless, it is a powerful technique for smoothing and peak force estimation.

6.4.3 Wiener Equalization

Because the harmonic Kalman Filter is unable to capture the high frequency content of the static model, an alternate approach is to use Wiener equalization. The advantage of the Wiener technique is that the filter can be formulated in the time domain. Wiener filters play a central role in a wide range of applications such as linear prediction, echo cancellation, signal restoration, channel equalization, and system identification [10]. The coefficients of a Wiener filter are calculated to minimize the average squared distance between the filter output and a desired signal. Here we can think about the Wiener filter from the classical perspective of “channel equalization”. With the typical terminology, the desired signal we wish to measure is distorted by the *channel* (the milling system compliance) and measured in noise. The goal of our filter is to recover the *transmitted signal* (the static force).

The Wiener filter is formulated by considering the case where the observation signal $y(m)$ is a distorted, noisy version of a transmitted signal $x(m)$ [13]. We wish to recover an estimate of $x(m)$ from $y(m)$ using an FIR filter of order P. Other filter structures could also be used to develop the filter, but a moving average (MA) FIR structure should work well in this application because it will naturally compensate for the system resonance well-modeled by the AR process (i.e. zeros do a good job of attenuating the resonance of system poles).

The FIR estimate of the static force profile is given by

$$\hat{x}(m) = \sum_{k=0}^P w_k \cdot y(m - k) \quad (6.9)$$

Where w is the vector of optimal FIR coefficients, and y is the vector of measurements. In vector notation, this estimate is given by

$$\hat{x}(m) = \mathbf{w}^T \mathbf{y}(m) \quad (6.10)$$

The typical Bayesian estimation procedure suggests that we should define a cost function, and then find the solution which minimizes the expected value of the cost (i.e. minimizes the Bayesian risk). In this case, we define the cost as the square of the estimation error. Our job now is to find the set of filter coefficients, w_k which minimize the expected value of the estimation error.

$$\begin{aligned} \text{Bayesian risk} &= E[x(m) - \hat{x}(m)]^2 \\ &= E \left[x(m) - \sum_{k=0}^P w_k \cdot y(m-k) \right]^2 \end{aligned} \quad (6.11)$$

If we assume that $x(m)$ and $y(m)$ are ergodic, then minimizing the expected value is the same as minimizing the time average value. Thus, for N observations, we can approximate the ideal solution by finding the set of filter coefficients which minimize the summed square error (SSE):

$$\begin{aligned} SSE &= \sum_{m=0}^{N-1} e^2(m) = \sum_{m=0}^{N-1} [x(m) - \hat{x}(m)]^2 \\ SSE &= \sum_{m=0}^{N-1} \left[x(m) - \sum_{k=0}^P w_k \cdot y(m-k) \right]^2 \end{aligned} \quad (6.12)$$

Which is minimized by taking the partial derivative with respect to the j^{th} coefficient

$$\begin{aligned} \frac{\partial SSE}{\partial w_j} &= \sum_{m=0}^{N-1} 2 \left[x(m) - \sum_{k=0}^P w_k \cdot y(m-k) \right] [-y(m-j)] \\ \frac{\partial SSE}{\partial w_j} &= -2 \left[\sum_{m=0}^{N-1} x(m)y(m-j) - \sum_{k=0}^P w_k \sum_{m=0}^{N-1} y(m-k)y(m-j) \right] \end{aligned} \quad (6.13)$$

Equation 6.13 can be simplified by recognizing that the summations are linear transforms of the biased definition of the sample auto-correlation and sample cross-correlation functions. Thus,

$$\sum_{m=0}^{N-1} x(m)y(m-j) = N \cdot \hat{r}_{yx}(j) \quad (6.14)$$

And

$$\sum_{m=0}^{N-1} y(m-k)y(m-j) = N \cdot \hat{r}_{yy}(k-j) \quad (6.15)$$

can be replaced in Equation 6.13 to obtain

$$\frac{\partial SSE}{\partial w_j} = -2 \left[\hat{r}_{yx}(j) - \sum_{k=0}^P w_k \cdot \hat{r}_{yy}(k-j) \right] \quad (6.16)$$

which is minimized by setting the partial derivative equal to zero.

$$\begin{aligned} \frac{\partial SSE}{\partial w_j} = -2 \left[\hat{r}_{yx}(j) - \sum_{k=0}^P w_k \cdot \hat{r}_{yy}(k-j) \right] &= 0 \\ \sum_{k=0}^P \hat{r}_{yy}(k-j) w_k = \hat{r}_{yx}(j), \quad j = 0, 1, 2, \dots, P & \quad (6.17) \end{aligned}$$

Thus we have P+1 equations, and P+1 unknowns. In matrix notation, let

$$\begin{aligned} \hat{R}_{yy} &= \begin{bmatrix} \hat{r}_{yy}(0) & \hat{r}_{yy}(1) & \dots & \hat{r}_{yy}(P) \\ \hat{r}_{yy}(1) & \hat{r}_{yy}(0) & \dots & \hat{r}_{yy}(P-1) \\ \vdots & \vdots & \ddots & \vdots \\ \hat{r}_{yy}(P) & \hat{r}_{yy}(P-1) & \dots & \hat{r}_{yy}(0) \end{bmatrix} \\ \hat{r}_{yx} &= \begin{bmatrix} \hat{r}_{yx}(0) \\ \hat{r}_{yx}(1) \\ \vdots \\ \hat{r}_{yx}(P) \end{bmatrix} \\ \mathbf{w} &= \begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_P \end{bmatrix} \end{aligned} \quad (6.18)$$

Finally, we can determine the filter coefficients which minimize the summed square estimation error by using matrix inversion.

$$\begin{aligned} \hat{R}_{yy} \mathbf{w} &= \hat{r}_{yx} \\ \mathbf{w} &= \hat{R}_{yy}^{-1} \hat{r}_{yx} \end{aligned} \quad (6.19)$$

If the signals are ergodic, then the ideal solution which minimizes the expected value of the estimation error squared (rather than the time averaged value) can be found in the limit as the number of observations approaches infinity [8]. This is the ideal Wiener solution for recovering a signal using an FIR filter. The conventional least squares solution to the Wiener filtering problem [8, 9] is similarly formulated as

$$\begin{aligned}
 e(m) &= x(m) - \hat{x}(m) \\
 &= x(m) - \sum_{k=0}^P w_k y(m-k), \quad \text{for } m = P \dots N-1-P
 \end{aligned} \tag{6.20}$$

Which in matrix notation, can be expressed as

$$\begin{bmatrix} e(0) \\ e(1) \\ e(2) \\ \vdots \\ e(N-1) \end{bmatrix} = \begin{bmatrix} x(0) \\ x(1) \\ x(2) \\ \vdots \\ x(N-1) \end{bmatrix} - \begin{bmatrix} y(0) & y(-1) & y(-2) & \dots & y(1-P) \\ y(1) & y(0) & y(-1) & \dots & y(2-P) \\ y(2) & y(1) & y(0) & \dots & y(3-P) \\ \vdots & \vdots & \vdots & \dots & \vdots \\ y(N-1) & y(N-2) & y(N-3) & \dots & y(N-P) \end{bmatrix} \cdot \begin{bmatrix} w_0 \\ w_1 \\ w_2 \\ \vdots \\ w_{P-1} \end{bmatrix} \tag{6.21}$$

$$e = x - Y \cdot w$$

We can minimize $e^T e$ by setting

$$w = (Y^T Y)^{-1} Y^T x \tag{6.22}$$

This is the standard least squares solution for minimizing the residual error squared for an over-determined set of linear equations ($N > P$). This matrix formulation is identical to the Wiener solution for a finite set of observations, because both formulations minimize $e^T e = \text{SSE}$.

We can now take a look at how this method works for our milling data. Again, the goal is to develop the optimal FIR filter coefficients that best recover the “static force” which results from the static chip thickness. The Wiener method requires a “training signal”, $x(m)$, which is used to determine how the desired signal is different from the observation sequence. This is extremely powerful, because we can use the infinitely-stiff static force model to train the filter to reject vibrations. Figure 6.35 shows the simulated static force profile superimposed on the measurement which has been corrupted by the system dynamics.

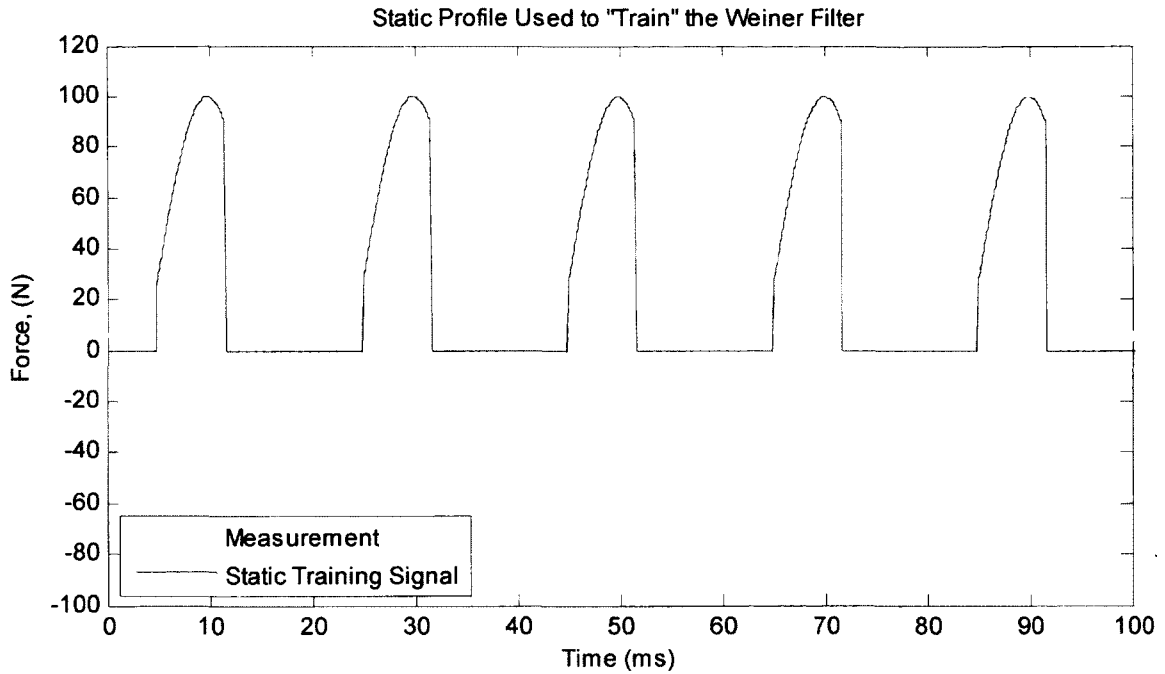


Figure 6.35 - The static model is used as a training signal to determine Wiener pole locations

Determining the optimal filter coefficients is a simple matter of sorting observations into the appropriate matrix locations, as dictated by Equation 6.21. The only difficult question is determining how to specify model order. This design choice, as always, is aided by knowledge of the spectrum. Our Linear Prediction models showed that the resonance in the spectrum is well-modeled by three spectral peaks. For this reason, choosing $P=7$ is a reasonable design choice. This 7th order polynomial in z will have 6 complex roots corresponding to three spectral nulls (Figure 6.36).

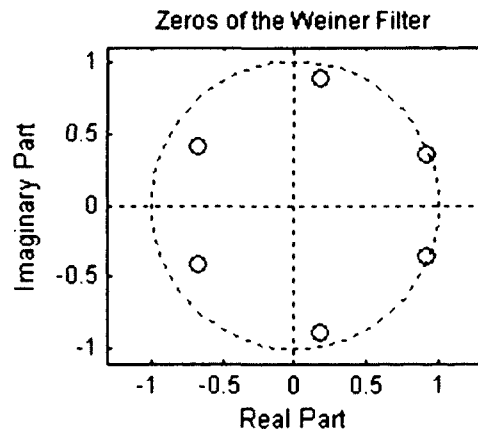


Figure 6.36 - Z-Plane of the Wiener filter model

The frequency response of our Wiener FIR equalization filter is easily derived from the filter structure. Recall that our estimate of the static force is given by

$$\hat{x}(m) = \sum_{k=0}^P w_k y(m-k) \quad (6.9)$$

Taking the Z-transform of both sides, we obtain

$$\hat{X}(z) = Y(z) \sum_{k=0}^P w_k z^{-k}, \text{ therefore } \frac{\hat{X}(z)}{Y(z)} = H(z) = \sum_{k=0}^P w_k z^{-k} \quad (6.23)$$

Thus, the frequency response of our FIR Wiener filter is obtained by letting $z = e^{j\omega} = e^{j2\pi fT}$, as shown in Figure 6.37 to filter the measurement signal shown in Figure 6.35.

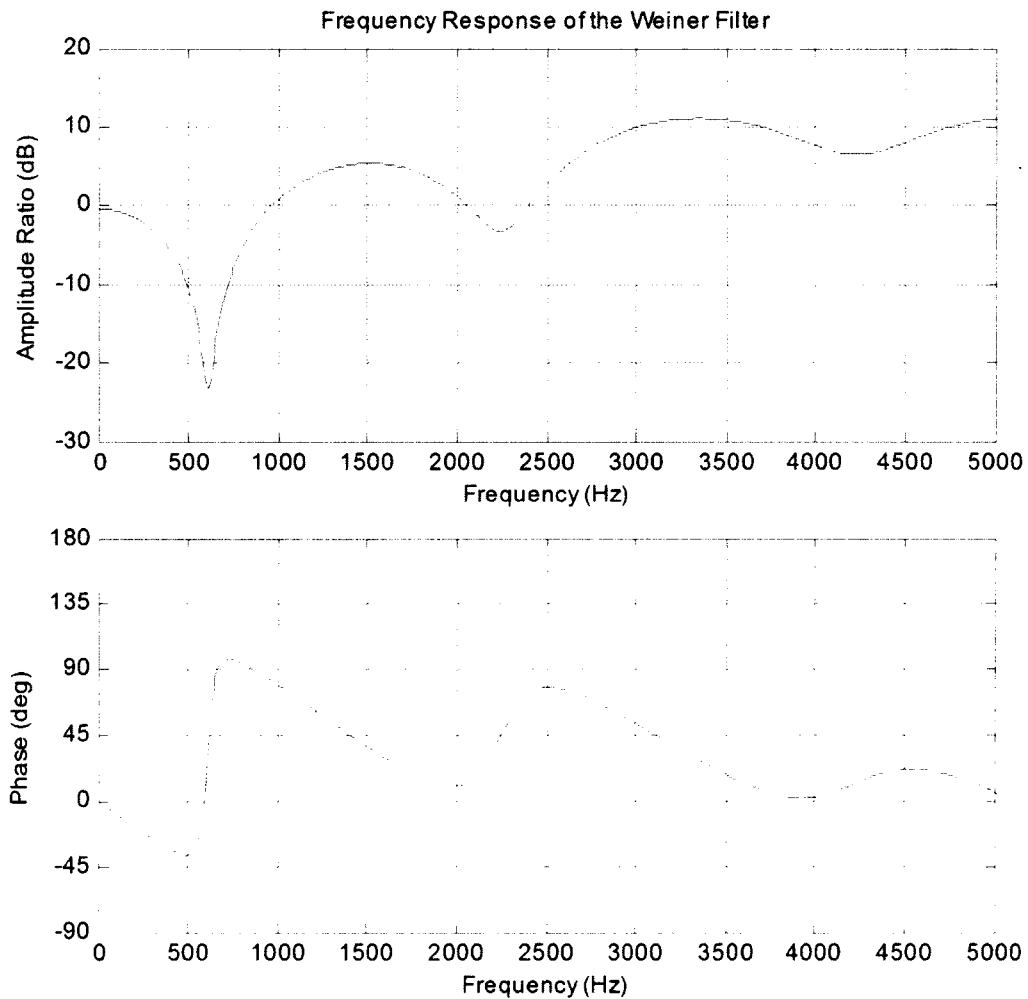


Figure 6.37 - Frequency response of the optimal Wiener solution

We see that that our 7th order FIR filter is designed such that its coefficients create spectral nulls at exactly the modes of resonance seen in our Linear Prediction model. This was accomplished by minimizing the SSE between the static training signal and the measured signal. This is a particularly neat technique because the filter zeros are determined from the data itself! (They are placed where there is poor coherence between the training signal and the measurement.) This is an outstanding result because the filter puts spectral nulls at exactly the modes of resonance seen in the measured strain signal.

When this filter is used on our measurement signal, the results are quite impressive.

Figure 6.38 shows the static signal estimate recovered from the measurement signal.

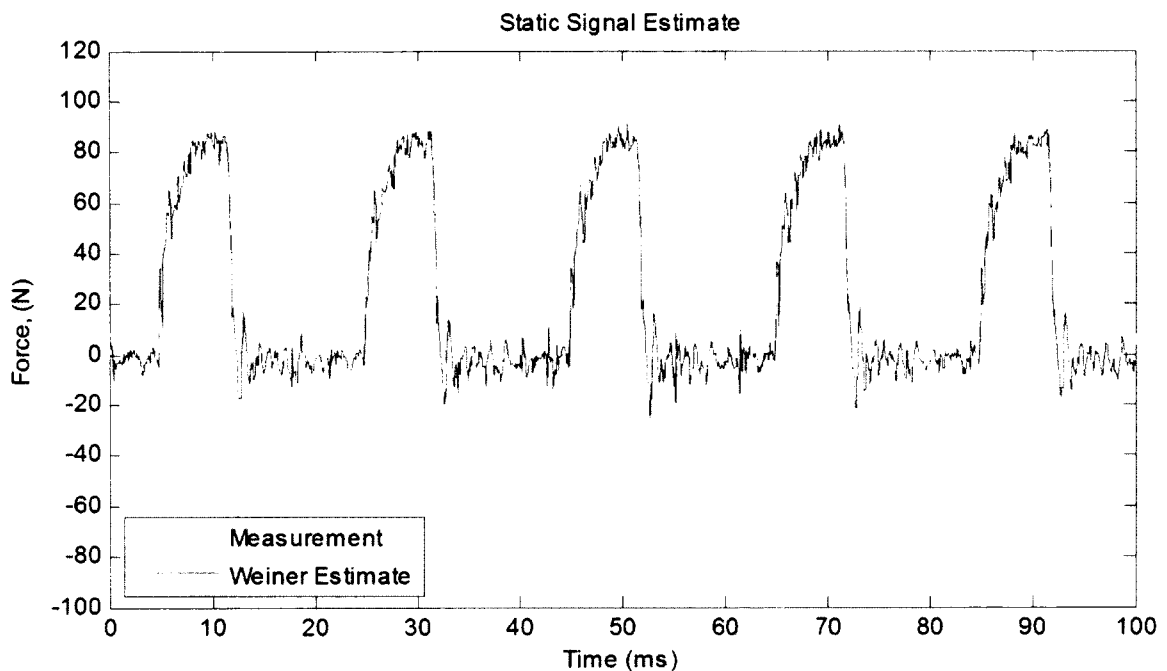


Figure 6.38 - Static signal estimate after applying the Weiner filter

The recovered static force estimate looks a lot like the infinitely-rigid training signal. Compared to the Kalman technique from above, our Weiner filter is clearly superior in terms of artificially extending the bandwidth because it does a better job of capturing the shape of the static profile. Recall that the Kalman estimate's peak is too rounded because the frequency domain model of the static profile ignores the high-frequency components. While the Kalman estimate is much

smoother than this Wiener estimate, this technique does not lose any of the sharpness in the shape of the static profile, because the model is trained in the time domain.

A 4-term FIR moving average filter can be used after the Wiener filter to smooth the profile, as shown in Figure 6.39. There is a trade-off between smoothing and phase shift, however, because a linear-phase MA filter has a delay of $(P-1)/2$ samples [19]. The sampling rate of the signal should be considered to select the model order for the moving average filter.

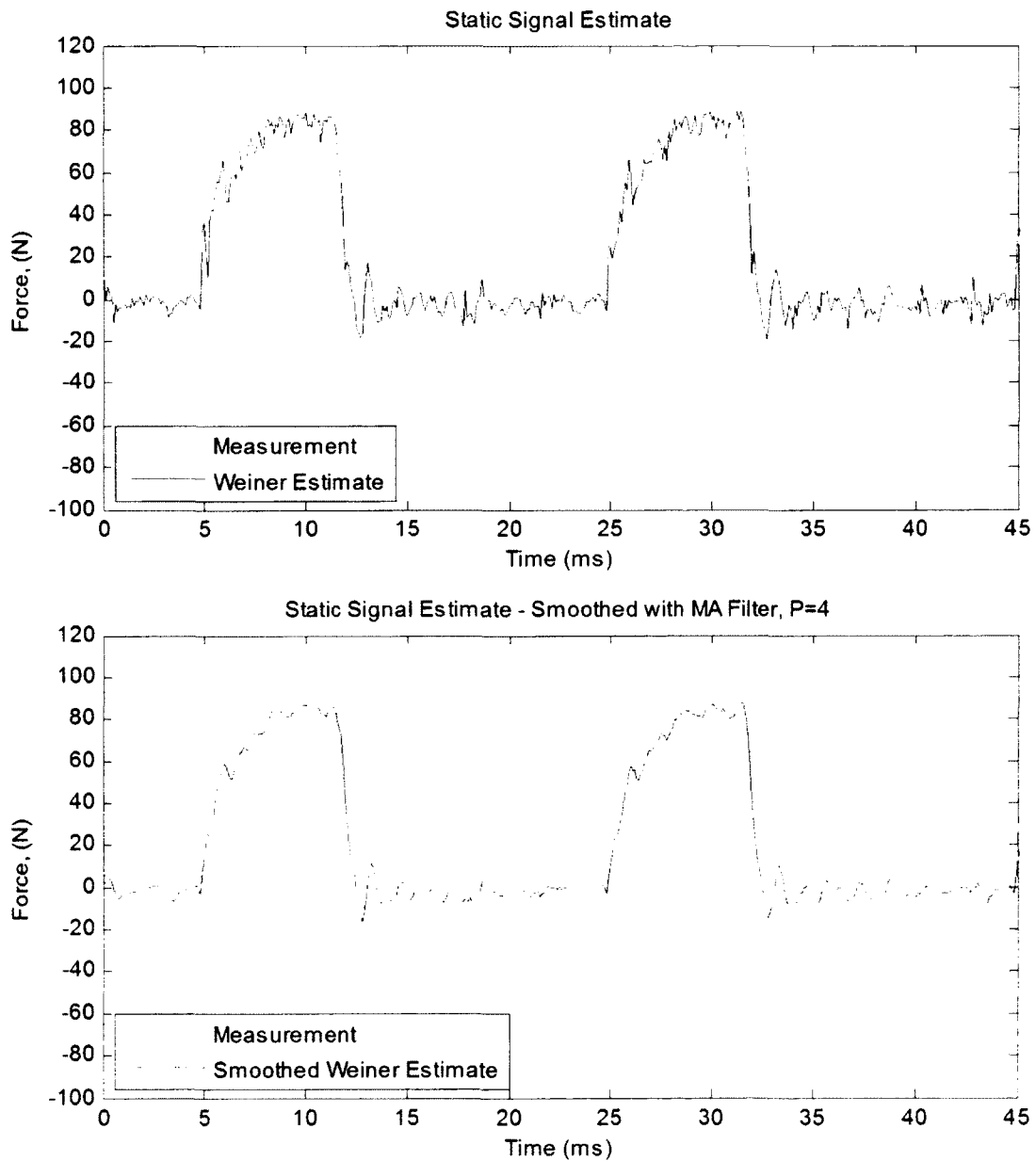


Figure 6.39 - Output of the Wiener filter can be smoothed with a low-order MA filter

6.5 Sub-Optimal Filtering for Signal Enhancement

Sub-optimal filtering techniques are often favored over optimal least-squares techniques for their comparable performance complemented by superior computational efficiency. This section discusses such techniques. Presented here is a non-optimal, model-based technique for estimating the applied cutting force followed by a discussion of how simple linear filters can be used to remove unwanted vibrations.

6.5.1 Dynamic Chip Load Filter

The notion of a “time-lagged difference sequence” was presented above, illustrating that knowledge of the spindle speed can be used to estimate the variation in force from cycle to cycle. This time-lagged difference sequence, or *variation sequence*, has a similar shape as the dynamic chip load because the measured strain signal is largely indicative of tool deflection. This assertion is justified by looking at the magnitude spectrum of the Linear Prediction model (Figure 6.22) where we see that most of the energy in the strain signal comes from the fundamental mode of vibration.

The power of this filtering technique is that it allows us to work around the problem of convolution and actually estimate the *applied cutting force*. This approach stands in contrast to all of the other filtering techniques discussed thus far, which were aimed at artificially increasing system bandwidth. As mentioned in the introduction of this chapter, we cannot use a linear filter structure (FIR, IIR) to estimate the actual cutting force because the cycle-delay feedback loop makes the dynamic behavior recursive.

If we are creative, however, we can exploit the fact that our strain-based sensor measures displacement to estimate the force that results from the dynamic chip thickness. We can do this by estimating the shape of the dynamic chip load in units of strain, scaling by the static sensitivity, and superimposing this measurement on the static force profile. This is shown schematically in Figure 6.40.

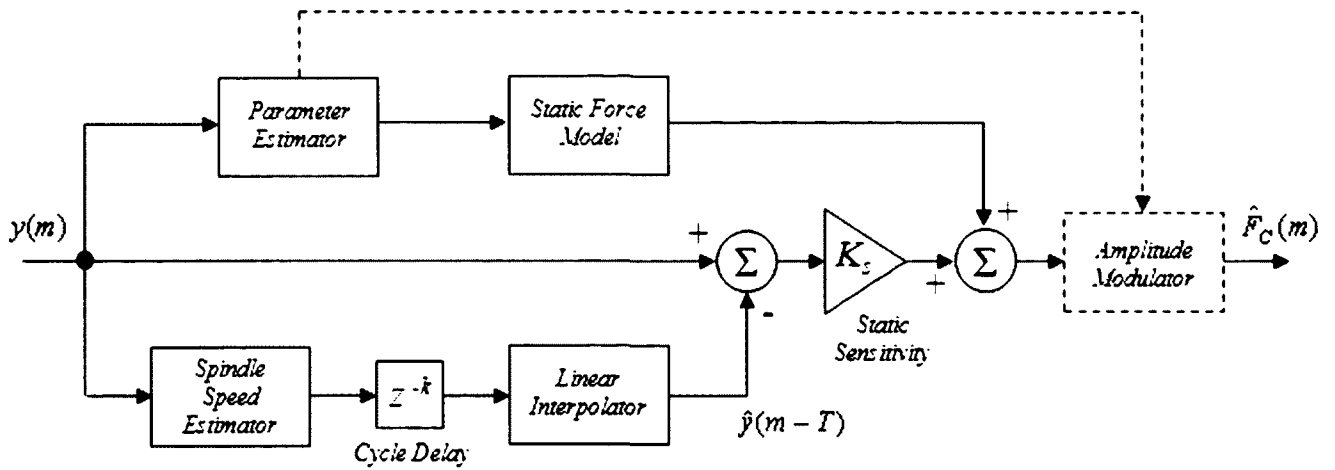


Figure 6.40 - Block diagram of the dynamic chip load filter

This process assumes that the static sensitivity accurately maps strain to force. Some sort of amplitude modulation may be required to compensate for gain as a function of frequency to accurately estimate force magnitudes. No work has been done to investigate how this amplitude modulation should be performed. Figure 6.41 shows the signals required to create the cutting force estimate. The measured force, $y(m)$, is differenced based on the period of revolution, to obtain the variation sequence given by $y(m) - \hat{y}(m - T)$. This signal is scaled by the static sensitivity, and superimposed on the static force profile.

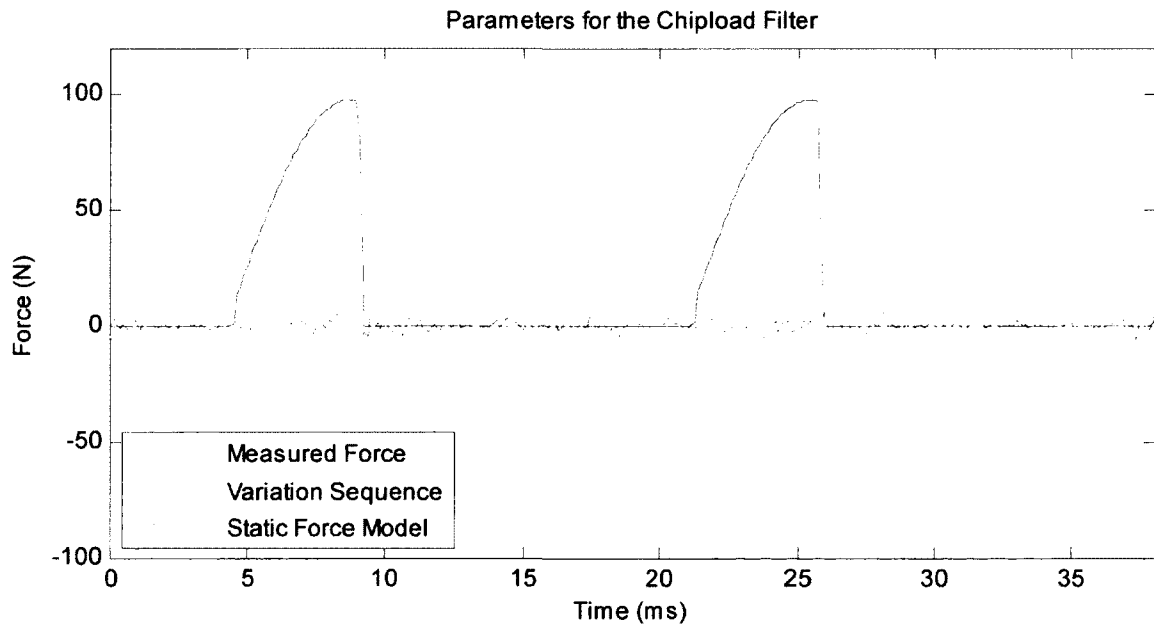


Figure 6.41 - Parameters for the dynamic chip load filter

Superimposing the force from the dynamic chip thickness on the static model results in our estimate of the actual cutting force, as shown in Figure 6.42.

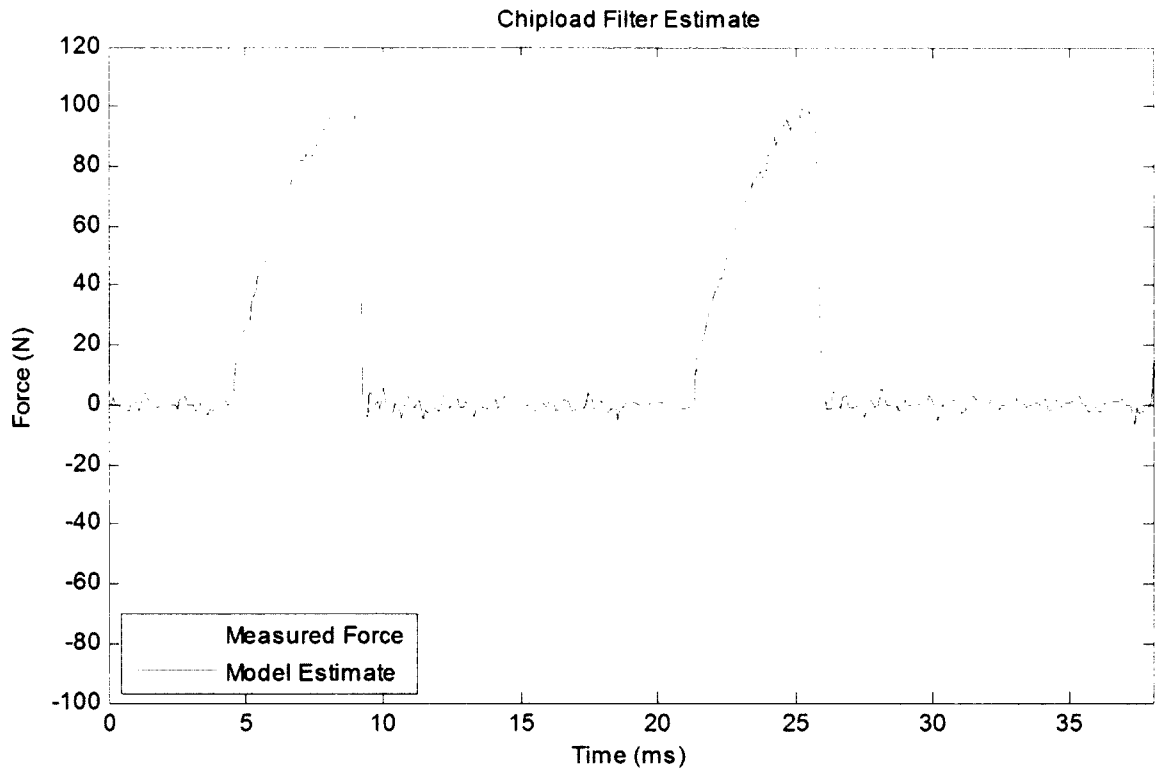


Figure 6.42 - Dynamic chip load filter estimate of the applied cutting force

For this stable cut, even though the measured force shows a lot of vibration, the actual cutting force looks a lot like the static chip thickness because variations are small. In this way, we can exploit our record of tool vibration (in units of force), to combine the force resulting from the dynamic chip thickness on a model-based static estimate.

Recall that our experimental validation in Chapter 5 showed that peak forces were in general agreement with those of the Kistler dynamometer, even for the higher spindle speeds. This observation is a preliminary validation for simply scaling the time-lagged strain sequence by the static sensitivity, without considering frequency-dependent amplitude modulation. This should be investigated as future work.

6.5.2 Linear Time-Invariant Filters

Our discussion of adaptive filtering and model-based filtering would be incomplete without comparing the relative performance of these estimators to that of a simple linear time-invariant notch filter. With the goal of artificially extending the bandwidth of the sensor, intuition tells us that a notch filter should do a reasonable job of removing vibration because most of the resonant energy is contained in the fundamental.

To compare estimators, a notch filter was designed using the Matlab built-in function “IIRnotch.m”. As previously shown in Figure 6.24, the fundamental mode of vibration is seen to move with some uncertainty. For this reason, the center of the notch was placed at the mean of the in-cut fundamental frequency (as informed by LPC), and the half-power width of the notch was specified by the spread of the data. The IIR filter design is shown below in Figure 6.43.

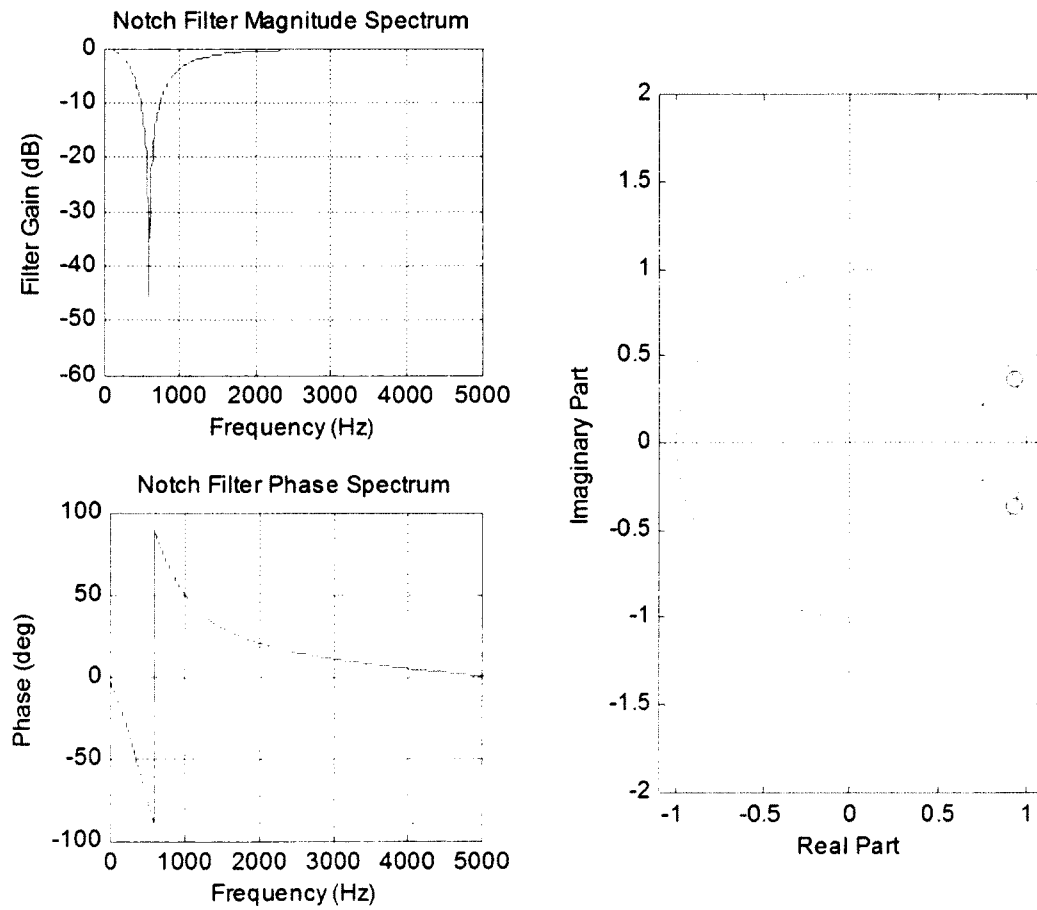


Figure 6.43 - Design parameters for the IIR notch filter

The frequency response is obtained by evaluating the Z-transform of the filter by the variable substitution, $z = \exp(j\omega)$. The application of this filter on the strain signal is shown in Figure 6.44.

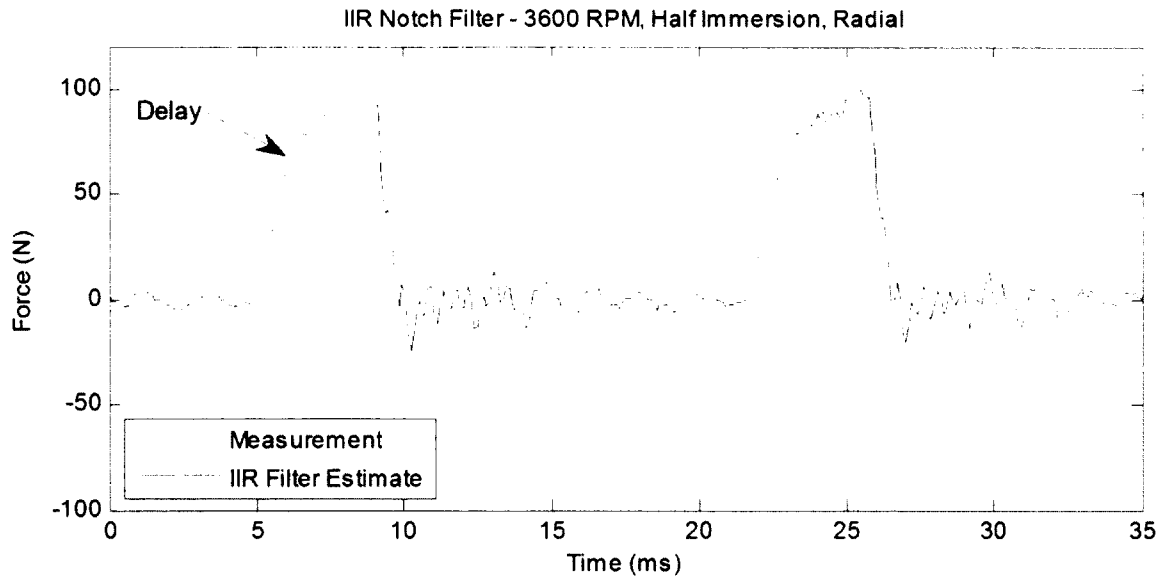


Figure 6.44 - Implementation of the IIR notch filter

We see a small time delay in the filtered response, but this simple linear filter does a great job of removing most of the vibration. Typically, we like to implement LTI filters that have linear phase to prevent phase distortion, so we implement the IIR notch filter with forward-backward (zero-phase) filtering for comparison (Figure 6.45).

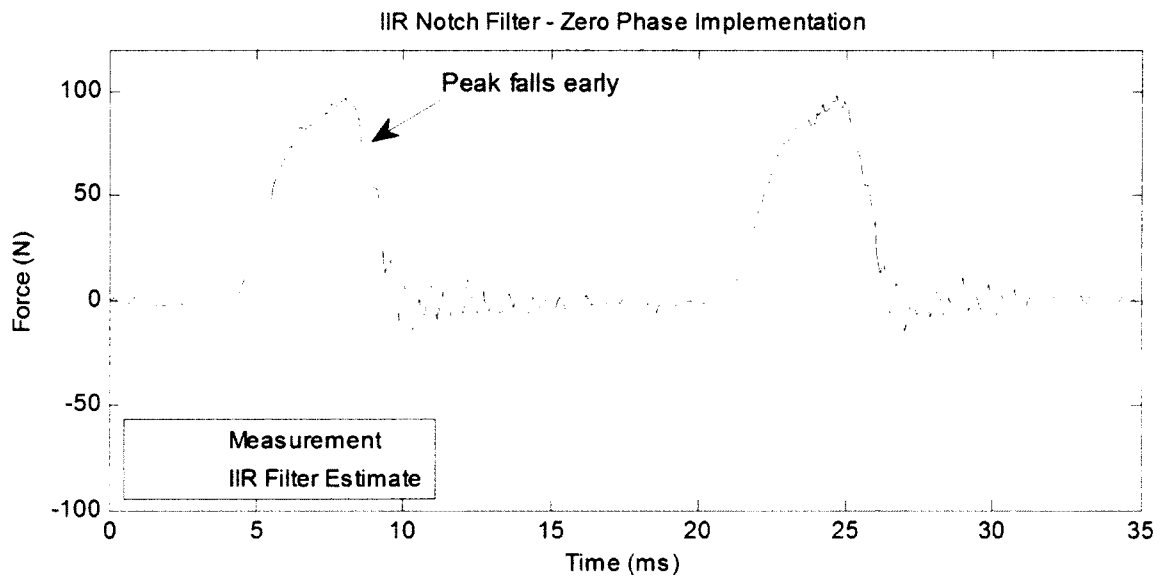


Figure 6.45 - Zero-phase implementation of the IIR notch filter

The interesting result we see in the response of the zero-phase implementation of the IIR notch filter (Figure 6.45) is that the peak falls much earlier than it did with the forward implementation. This is interesting because the phase distortion of the notch filter actually *helps* the filtered signal to look more like the static force estimate we are trying to recover.

If we wanted to implement a notch filter using an FIR structure, we would need to design an FIR filter with approximately the same impulse response as the IIR filter. This is easily accomplished by truncating the impulse response of the IIR filter, and using these values as the filter coefficients [19], as seen in Figure 6.46.

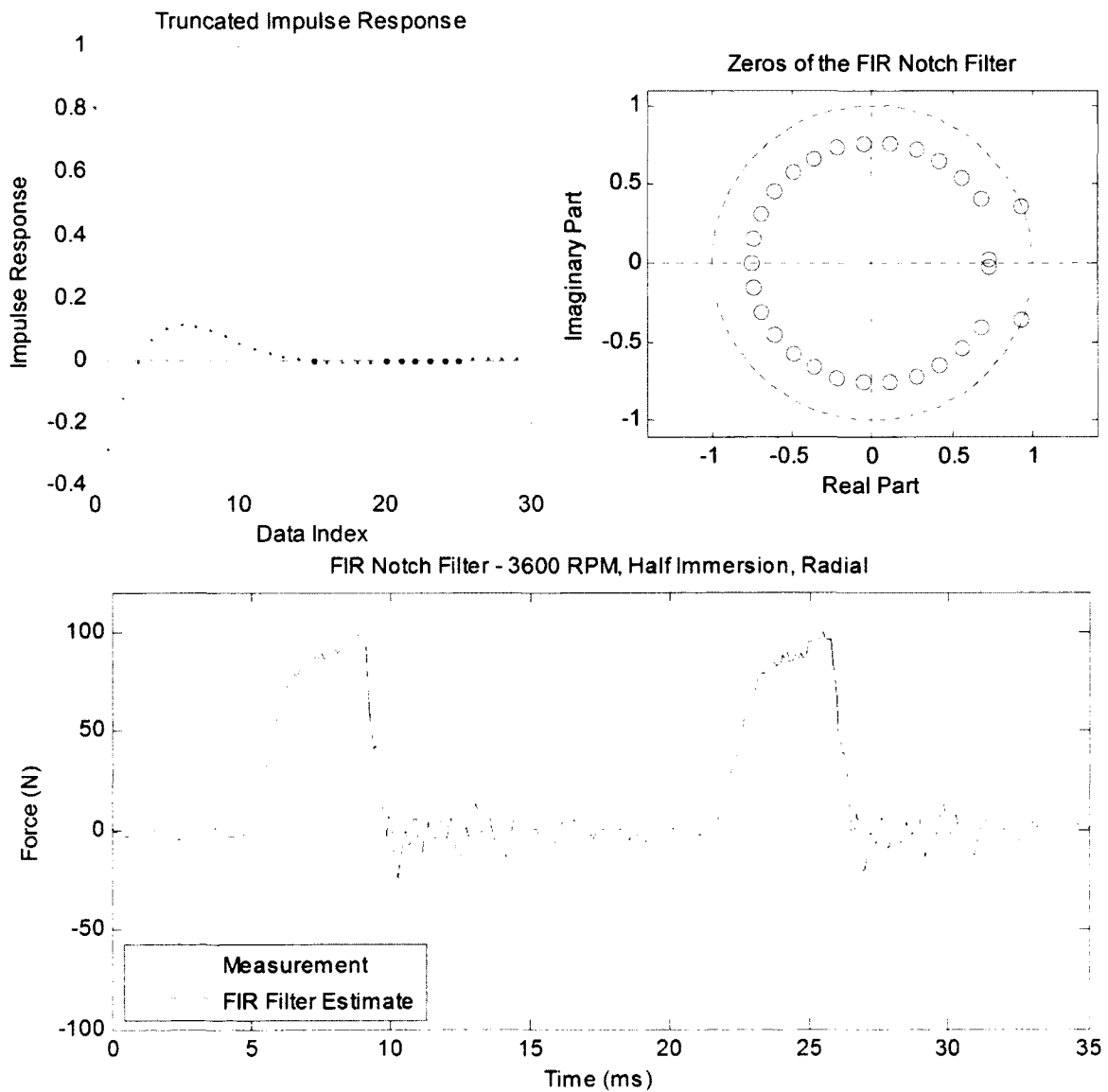


Figure 6.46 - FIR notch filter implementation

We see no noticeable difference in the static force estimate using the FIR notch filter implementation from that of the IIR model. The advantage of the IIR model is that it uses fewer sums and multiplies because it only has 2 poles and 2 zeros, as opposed to this FIR filter which has 30 zeros.

6.6 Summary

This chapter has shown that Linear Predictive Coding is capable of estimating the system resonance for both open-loop and closed-loop vibrations. This is an exceedingly powerful tool for system identification because it allows us to inform an adaptive filter of the system vibrations cycle by cycle. Real-time implementation would require sufficient processing overhead to implement the LPC algorithm. This should be investigated as future work.

Optimal model-based filters were also developed to artificially increase the bandwidth of the sensor. While the Kalman filter and the Weiner filter both have their merits, Weiner equalization is seen to be particularly useful. It is capable of removing vibrations without a time delay, while simultaneously maintaining the high frequency content of the signal necessary to preserve the sharp corners of the static profile. The Kalman technique may be able to achieve better results if a different model is used. While the harmonic model loses the high frequency content needed to maintain the sharp corners of the static profile, it may be possible to build an equalization model via linear prediction. This should be investigated as future work.

Sub-optimal techniques were also shown to be useful. The dynamic chip load filter exploits the physics of the end milling system to estimate the applied cutting force. This is different from the other techniques which simply aimed to artificially extend the bandwidth by approximating the static profile.

Notch filters were also shown to remove vibrations well. These simple filters are well suited for applications when the corresponding time delay is unimportant. For example, using a notch filter on the measured signal before the Altintas average force calibration method is employed

will probably provide good results. The time delay is a problem, however, if we wish to use the filtered result to subtract the “static force” and obtain the compliant residual measurement.

Table 6.1 provides a summary of the estimators considered in this chapter and their possible applications.

Table 6.1 - Comparison of estimators useful for signal processing

	Linear Prediction	Kalman Filter	Weiner Filter	Dynamic Chip Load Filter	Notch Filter
Useful Applications	System identification	Artificially extending the bandwidth	Artificially extending the bandwidth	Estimating applied cutting force	Artificially extending the bandwidth
Advantages	Least-squares technique Describes the system well Track in-cut and out-of-cut variation	Least-squares technique No phase delay Very robust	Least-squares technique No phase delay FIR model is simple	Estimates applied force, not static force Physics-based model	Effective at removing vibration Simple Can implement in real time
Disadvantages	Need method of coherent triggering to mitigate variance of estimates	Requires much processing Smoothing and smearing of peak force	Filter requires training	Very sensitive to errors in spindle speed measurement	Small time delay
Possible for Real Time?	Maybe Need method of coherent triggering for indexing	No Too much computation involved	Maybe Training signal necessary	Maybe Accurate spindle speed must be known	Yes Build a wide notch to account for variation

CHAPTER 7

CONCLUSIONS AND FUTURE WORK

7.1 Conclusions

Through this work, the Smart Tool was shown to achieve its fundamental design goal of accurately resolving bending strains from a combined loading scenario while remaining insensitive to unwanted components of strain. To this end, the sensor is capable of resolving static loads with a total measurement uncertainty of less than 3 percent full scale while maintaining an effective resolution of 4.5 N and a span of approximately 1400 N. A summary of the significant experimental results from static calibration (Chapter 3) are repeated below in Table 3.3.

Table 3.3 (Repeated) – Summary of experimental results from static calibration

Attribute	Specification	Experimental Result	Specification Met?
Effective Resolution	4.5 N minimum	4.51 N	Yes
Span	1330 N minimum	1395 N	Yes
DC Stability	< 3% full scale	Drift < 2% full scale	Yes
Bending Crosstalk	< 1% full scale	Radial: 0.506 %	Yes
		Tangential: 2.68 %	No
Torsional Crosstalk	< 1% full scale	Radial: 0.37 %	Yes
		Tangential: 0.15 %	Yes
Total Error	< 5% full scale	< 3% full scale	Yes

As shown in Chapters 4 and 5, the dynamic performance of the sensor is less than ideal because large tool vibrations degrade the sensor's ability to accurately measure force. The fundamental natural frequency of the Smart Tool is approximately 630 Hz, however, this mode is seen to change depending on the boundary condition at both the cutting tip and at the spindle. For example, Figure 4.5 shows that there is variation in these parameters with respect to spindle speed, and Figure 6.23 shows that these parameters also changed in-cut versus out-of-cut for a fixed spindle speed of 3,000 RPM. Such a low natural frequency is a problem for displacement-based strain sensors like our Smart Tool because vibrations become convolved with the desired measurement thus degrading our ability to accurately measure the applied cutting force. Furthermore, because the vibrations overlap with the signal of interest in the frequency domain, the signal cannot be separated from the "noise" (i.e. distortion from system dynamics) with a simple low pass filter.

Several techniques in signal processing were presented to artificially extend the bandwidth of the sensor (*Kalman filter, Weiner filter, notch filters*), and a model-based technique called the *dynamic chip load filter* was developed to estimate the applied cutting force by interpreting the shape of the strain signal as tool deflection. Furthermore, *Linear Predictive Coding* was shown to be a powerful tool for dynamic parameter identification of system resonance. Linear prediction is particularly intriguing because it allows for the development of an *adaptive* filtering technique rather than using a time-invariant filter which merely corrects for the signal distortion on average. As shown in Chapter 6, these techniques can be used to circumvent problems associated with low bandwidth by modeling and removing forced vibrations.

Chapter 5 showed that the Smart Tool's ability to measure force in a dynamic environment is comparable to that of the Kistler 3-axis force dynamometer. For the 600 RPM experimental validation, both sensors are shown to accurately measure the cutting force and yield nearly identical measurements of force. At higher spindle speeds, again, the two sensors are in good agreement about where tooth engagement starts and stops, and they yield nearly identical

peak force measurements; at these higher spindle speeds, however, both sensors' measurements of force are distorted by sensor dynamics for the reasons described above.

This experimental validation (Chapter 5) shows that our sensor is both accurate and useful for measuring cutting forces in milling. The advantage of our sensor, as opposed to the Kistler, is that the Smart Tool is less invasive to the machining process. Furthermore, while both sensors' measurements are seen to be corrupted by sensor dynamics, the vibrations in the Smart Tool's strain signal are largely indicative of tool deflection. This is important because tool deflection is usually the largest contributor to the total system compliance. Thus, with an indication of the strain resulting from tool deflection, we can reconstruct the applied cutting force by assuming that this force is proportional to the dynamic chip thickness (Section 6.5.1).

While our sensor's design was shown to meet its design objectives, there are some disadvantages to the design architecture in its current state. Foremost, it is a problem that the sensor is limited to a single cutting tooth. While this restriction is necessary to measure bending strains (which is how we are able to meet the constraint for total measurement error), it makes the Smart Tool impractical for industrial applications other than finish cutting. Similarly, the Kistler is ill-suited for industrial applications because of its high cost and invasive nature. The Kistler has the advantage, however, that it works with all cutting tools. The Smart Tool has the additional disadvantage that the current data transmission board is only capable of transmitting one signal at a time.

Nonetheless, the Smart Tool is capable of capturing detailed information about the milling process at the tool tip. Even if a multi-tooth insert were used with this sensor design, the measured output is valuable when interpreted as tool deflection. This information could be used in a multiple-sensor smart machining system to measure tool vibrations and subsequently perform dynamic parameter estimation or spectral analysis to analyze stability of the current milling operation.

7.2 Extension of Signal Processing Techniques for Real-Time Implementation

The Smart Tool project exists to develop novel sensors that are ultimately useful for real-time quality control in milling. The natural extension of the work in this thesis is to consider how the sensor lends itself to real-time process improvement. Figure 7.1 shows a suggested block diagram for real time implementation of the Smart Tool in the framework of a smart machining system. The paragraphs below describe the motivation for interpreting the strain signal in the following way:

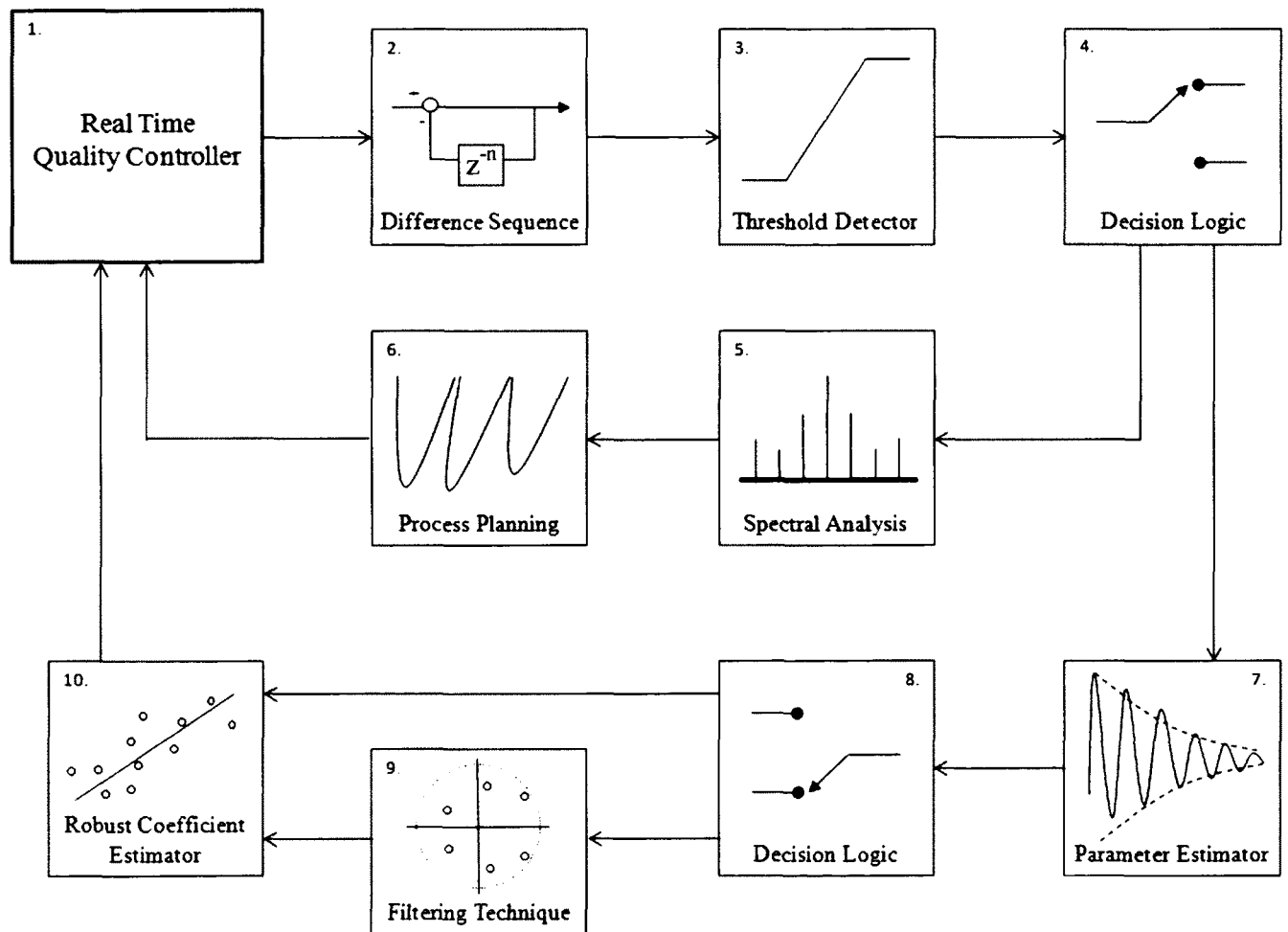


Figure 7.1 – Suggested block diagram for real-time implementation of the Smart Tool

Block 1 – Real Time Quality Controller

The real time quality controller uses cost and objective functions to meet process objectives. For example, during a roughing operation, the process objective is to remove material as efficiently as possible, which requires a tradeoff between feedrate, spindle speed, and rate of tool wear. Here, peak force measurements are important to monitor the process and prevent tool breakage. Alternatively, during a finish cut, it may be more important to maintain good surface quality, and spectral analysis could be employed to track vibrations.

Block 2 – Difference Sequence

With knowledge of the spindle speed, it is a simple matter to difference the measured strain signal to obtain a sequence that is proportional to the dynamic chip load. Essentially, this block finds the difference in force at the same angle of rotation for subsequent tooth passes. This is done first, because it is a computationally efficient way of assessing the stability of the cut. If vibrations remain in phase, the amplitude of this difference sequence is small; if vibrations become out of phase, this difference signal becomes large and the controller should take corrective action to maintain stability. For this reason, this time-lagged difference technique is especially useful for applications in chatter detection.

This block is performed first because it is a powerful tool for signal classification. Signal classification is necessary because the Smart Tool's measurement signal seems to fall into one of three categories at any given time:

1. Minimal vibration. Measured force is truly indicative of the applied cutting force.
2. Some vibration. Vibration is mostly repeatable from cycle to cycle. Here, the measured force is more indicative of tool deflection than it is of cutting force.
3. Lots of vibration. Vibration is not repeatable from cycle to cycle. The cut may be unstable.

The time-lagged difference sequence will be small for cases (1) and (2), but will not be small when vibrations become out-of-phase (3). Thus, by subtracting prior values of strain from the current value, we can make quick assessments about the stability of the cutting process. Accurately determining the spindle speed is a necessary intermediate step to implementing this algorithm. This is readily accomplished by using a Hall Effect sensor on the CNC spindle.

Block 3 – Threshold Detector

If cycle-to-cycle variations in measured strain are small, the cut appears to be stable. In this case, tool vibrations are either insignificant or highly repeatable, and we fall into signal classification category (1) or (2). If the strain signal becomes aperiodic, this will be reflected in the magnitude of the time-lagged difference signal making it possible to determine stability of the cut before damage is done to either the tool or the workpiece. A threshold can be established to determine if the cut may be unstable. Determining an appropriate threshold value should be investigated as future work. For example, time-lagged variations in measured force larger than 10 percent of the peak force may indicate that the cut is becoming unstable.

Block 4 – Decision Logic

The magnitude of the time-lagged difference sequence can be used to assess stability of the cut. This informs a logic switch as to what action should be taken. If the cut is potentially unstable, it is more important to find stable cutting conditions than it is to accurately know force magnitudes.

Block 5 – Spectral Analysis

Spectral analysis of the time-lagged sequence was shown in Chapter 6 to provide enhanced spectral estimation by completely removing the harmonics of the toothpassing frequency. This is a convenient tool for identifying potential chatter frequencies.

Block 6 – Process Planning

With knowledge of the spindle speed and potential chatter frequencies, it may be possible to find more stable cutting conditions before damage is caused to either the tool or the workpiece. Based on the block diagram in Figure 7.1, process planning is necessary here to find more stable cutting conditions at the onset of chatter.

Other important aspects of process planning include the tracking of tool wear, and performing in-situ adjustments in feedrate to compensate for tool wear. The quality controller must decide when it is time to replace the tool based on cost functions of several variables. The contents of this block should be further investigated as future work.

Block 7 – Parameter Estimator

If the cut is stable, the important question is whether or not vibrations are significant. Implementing the time-lagged difference sequence does not tell us whether vibrations are large or small – it only tells us that they are repeatable. Parameter estimation can be used to look at the damping of the open-loop vibrations. If the magnitude of the vibrations is large, filtering may be required on the strain signal before the quality controller calibrates the Altintas force model. This filtering may be necessary to smooth the force profile for successful calibration using the instantaneous method. Dynamic parameter estimation can also be used to inform this filter which frequency bands contain unwanted vibrations. Linear prediction has useful applications here as shown in Chapter 6.

Block 8 – Decision Logic

This logic switch uses knowledge of the magnitude of vibrations to assess whether or not filtering is required on the measured strain signal before calibration. Future work should investigate an appropriate magnitude threshold for open-loop vibration, below which no filtering of the strain signal is necessary. Integration of the power spectrum over narrow frequency bands can provide a measure of the energy of the vibrations.

Block 9 – Filtering Technique

Selection of an appropriate filtering technique depends on which method will be used to calibrate the Altintas force method. If the average-force method [1] for online calibration will be used, it may be possible to simply use a notch filter to remove the unwanted vibrations. If the instantaneous method is required for calibration because the average force matrices are ill-conditioned (due to lack of variation in the average chip thickness), it may be required to implement a filter with negligible phase delay. In this case, the Weiner filter is a superior choice, as shown in Chapter 6.

Block 10 – Robust Coefficient Estimator

A quality controller based on Altintas model coefficients [1] would use either the average force method or the instantaneous method to calibrate the force model. A force model is powerful because it gives the quality controller the ability to estimate the forces resulting from different cutting conditions. These model coefficients are then used to make changes in process variables, such as modifying the feedrate and/or spindle speed, or deciding that excessive tool wear requires replacement of the cutting inserts.

7.3 Suggested Topics and Direction for Future Work

This section presents suggested topics for future work in specific areas of sensor development and signal processing.

7.3.1 Sensor Design and Wireless Communications

Successful integration of the Smart Tool into a smart machining system requires a few modifications to the current sensor architecture. First and foremost, the data transmission board needs to be upgraded from a single-channel design to a two-channel design, thus allowing for both radial force and tangential force to be measured simultaneously. Second, data collection should be facilitated via C++ so that the sensor is capable of interfacing with the real time quality

controller. Accessing data through MATLAB is too slow to be effective in real time. Lastly, the Smart Tool's data stream should be synchronized with that of the Kistler and the spindle Hall Effect sensor. This last step requires some sort of clock synchronization for the wireless network.

To be useful in industrial applications, the sensor design needs to be extended so that it is capable of accurately measuring forces on a multi-tooth cutter. The most logical way to achieve this goal is to use torsional strain gages on the tool holder body. This bridge configuration has been investigated before [2, 6, 14], but these attempts used semiconductor gages with a small physical footprint (i.e. gage pattern). Because of this small gage pattern, these designs did not cancel bending strains adequately, and the measurement suffered from large amounts of crosstalk.

It is the notion of the author that this problem might be circumvented by using foil strain gages with a large-grid shear pattern. Because of the limited accuracy that is achievable when mounting strain gages by hand, a large grid pattern may be able to compensate for small misalignments and effectively mitigate problems associated with cross sensitivity. The torsional shear stress distribution that results from the applied cutting force is theoretically axis-symmetric around the tool holder. The bending strains are not axis-symmetric, thus a large grid pattern allows for a larger percent overlap of the bridge elements. This overlap is the necessary condition to compensate for bending crosstalk in the presence of gage misalignment.

There is an inherent trade off in sensitivity by using this torsional design, however, because foil gages have significantly smaller gage factors, which is compounded by the fact that torsional strains resulting from the cutting force are much smaller than the bending strains (due to geometry of the cutting tool). The potential advantage of such a sensor, however, is that the design would be capable of using multiple cutting teeth. Also, tool holders are much stiffer in the torsional direction than they are in bending, so the sensor would also have a higher bandwidth, and filtering techniques may not be necessary.

7.3.2 Promising Techniques in Signal Processing

Many of the signal processing techniques presented in Chapter 6 showed significant merit to furthering the development of a real time quality controller. Other techniques may also provide valuable information to a real-time controller, or as useful techniques for post-processing in the lab. Suggested directions for future work are presented here.

1. **Linear Predictive Coding** should be further investigated as a primary tool for model building. A case study should be performed to evaluate uncertainty in the estimates of resonant frequencies as determined by the LPC poles. Coherent averaging was used to align cycles to improve estimates (See Appendix E). The sensitivity of AR pole estimates to choices in data indexing should also be investigated as future work.
- **Blind deconvolution** should be investigated as a tool for artificially extending the bandwidth of the sensor. It may be computationally more efficient than any of the other model-based filtering methods (Kalman filter and Weiner filter). Formulations for blind deconvolution based on linear prediction models or from knowledge of the input power spectrum can be found in [8].
 - **Frequency domain calibration** of the Altintas cutting force model would be a powerful tool because it completely eliminates problems with knowing the instantaneous angle of the cutting tooth. While nothing on this subject was formally included in this thesis, much time was spent working on frequency domain algorithms – unfortunately this work has not been wholly successful. Notwithstanding, it is the opinion of the author that knowledge of the spectrum could be used in a creative way to calibrate Altintas model coefficients in the frequency domain.

LIST OF REFERENCES

- [1] Altintas, Y., 2000, *Manufacturing Automation: Metal Cutting Mechanics, Machine Tool Mechanics, Machine Tool Vibrations, and CNC Design*, Cambridge University Press, UK. Print.
- [2] Suprock, C., 2011, *Dissertation: Wireless Sensor Integrated Tool for Characterization of Machining Dynamics in Milling*, Doctoral Dissertation, University of New Hampshire. Print.
- [3] Xu, M., Jerard, R. B., and Fussell, B. K., 2007, "Energy Based Cutting Force Model Calibration for Milling," *Computer-Aided Design and Applications*, 4(1-6), pp. 341-351.
- [4] Jerard, R. B., Fussell, B. K., Xu, M., and Cui, Y., 2006, "Process Simulation and Feedrate Selection for Three-axis Sculptured Surface Machining," *International Journal of Manufacturing Research*, 1(2), pp. 136-156.
- [5] Figliola, R., 2006, *Theory and Design for Mechanical Measurements*, 4th ed., John & Wiley Sons, Inc., USA. Print.
- [6] Jerard, R., Fussell, B., Suprock, C., Cui, Y., Nichols, J., Hassan, R., Esterling, D., 2009, "Integration of Wireless Sensors and Models for a Smart Machining System," *Proceedings of the ASME 2009 Manufacturing Science and Engineering Conference*.
- [7] Vishay Micro Measurements, Tech Note TN-512-1: *Plane-Shear Measurement with Strain Gages*. < http://www.intertechnology.com/Vishay/pdfs/TechNotes_TechTips/TN-512.pdf>.
- [8] Vaseghi, Saeed. *Advanced Digital Signal processing and Noise Reduction*. 3rd ed. Chichester, England: John Wiley & Sons, Ltd., 1996. Print
- [9] Proakis, John, G., and Dimitris Manolakis. *Digital Signal Processing*. 2nd ed. Upper saddle River, New Jersey: Pearson Education, Inc., 2007. Print.
- [10] Box, George E. P., Gwilym M. Jenkins, et al. *Time Series Analysis: Forecasting and Control*. 4th ed. Hoboken, New Jersey: John Wiley & Sons, Inc., 2008. Print.
- [11] Stengel, Robert F. *Optimal Control and Estimation*. 2nd ed. New York, New York: Dover Publications, Inc., 1994. Print.
- [12] Gelb, Arthur. *Applied Optimal Estimation*. 10th ed. Cambridge, Massachusetts: The M.I.T. Press, 1988. Print.
- [13] Brown, Robert G., and Patrick Y.C. Hwang. *Introduction to Random Signals and Applied Kalman Filtering with MATLAB Exercises and Solutions*. 3rd ed. New York, New York: John Wiley & Sons, Inc., 1997. Print.
- [14] Nichols, Jeffery., 2009. Master's Thesis: Design and Application of a Wireless Torque Sensor for CNC Milling. University of New Hampshire. Print.

- [15] “Modal testing Using a Hammer and Accelerometer”, HP 35670A Supplemental Operator’s Guide, Section 11, pp. 25-29.
- [16] Agilent 35670A Operator’s Guide, “*Chapter 2, Measuring Structures.*” Print.
- [17] Application Note 243-3, “The Fundamentals of Modal Testing”, Agilent Technologies. <<http://cp.literature.agilent.com/litweb/pdf/5954-7957E.pdf>>.
- [18] Tyler, Christopher T., Schmitz, Tony L., “*Process Damping Analytical Stability Analysis and Validation.*” Proceedings of NAMRI/SME, Vol. 40, 2012.
- [19] Lyons, Richard G. Understanding Digital Signal Processing. 2nd ed. Upper saddle River, New Jersey.: Prentice Hall. Pearson Education, Inc., 2004. Print.

APPENDIX A

EXPERIMENTAL PROTOCOLS FOR STATIC CALIBRATION

Static calibration provides a means of characterizing instrument sensitivity and cross-sensitivity to known inputs. The sensor must be able to measure tangential and radial forces for a single-tooth cutter, thus the sensitivity for both radial and tangential force measurements must be characterized. Unwanted components of strain from interfering inputs must either be shown to be negligible, or they must be removed through modeling and post processing.

Static calibration consists of five sets of experiments:

1. Calibration of bending sensitivity
2. Calibration of bending crosstalk
3. Calibration of cross-sensitivity to torsion
4. Calibration of axial sensitivity
5. Characterization of DC stability

Experiment 1 – Calibration of Bending Sensitivity

Objective

The purpose of this experiment is to characterize the sensitivity, span, and resolution in both the radial and tangential directions.

Equipment List

- C-channel Instron Fixture
- Instron servo-hydraulic 55s
- Load cell
- 2 point contact
- Rapid-prototyped angular alignment block
- Level
- Smart Tool v.10
- Laptop with Bluetooth capability and MATLAB

Experimental Protocol

Load Cell Calibration

The calibration file must be loaded for the appropriate load cell. Use the drop down menu bar for the Instron software application to load the appropriate calibration file.

Smart Tool Calibration of Bending Sensitivity

1. Attach the U-channel fixture to the Instron machine
2. Attach the Smart Tool to the fixture to calibrate the radial bending sensitivity
 - a. Tighten all bolts to 25 in-lbf
 - b. Record the distance from the gages to the point of contact
 - i. Measure this distance 10 times to assign a statistical confidence to the lever arm measurement.
 - c. Load and unload the tool to 300 lbf 10 times to work out any initial hysteresis
 - d. Load the tool from 0 lbf to 200 lbf in roughly 10 lbf increments
 - e. Unload the tool from 200 lbf to 0 lbf in roughly 10 lbf increments
 - f. Repeat this loading four more times for a total of five loading cycles
 - i. Record all measurements from the Instron load cell
 - ii. Record all measurements from the Smart Tool using the sth10live.m file with a moving window of 20 seconds.
3. Attach the Smart Tool to the fixture to calibrate the tangential bending sensitivity
 - a. Tighten all bolts to 25 in-lbf
 - b. Record the distance from the gages to the point of contact
 - i. Measure this distance 10 times to assign a statistical confidence to the lever arm measurement.
 - c. Load and unload the tool to 300 lbf 10 times to work out any initial hysteresis
 - d. Load the tool from 0 lbf to 200 lbf in roughly 10 lbf increments
 - e. Unload the tool from 200 lbf to 0 lbf in roughly 10 lbf increments
 - f. Repeat this loading four more times for a total of five loading cycles
 - i. Record all measurements from the Instron load cell
 - ii. Record all measurements from the Smart Tool using the sth10live.m file with a moving window of 20 seconds.

Data Reduction

1. Plot Smart Tool output versus radial force over the calibrated range
 - a. Plot the least squares fit line
 - b. Plot the 95% confidence interval
 - c. Plot the 95% prediction interval
2. Plot Smart Tool output versus tangential force over the calibrated range
 - a. Plot the least squares fit line
 - b. Plot the 95% confidence interval
 - c. Plot the 95% prediction interval
3. Determine the range of the sensor from the resolution and the span of the A/D converter

Characterized Parameters

- Load cell sensitivity and measurement confidence
- Smart Tool static bending sensitivity in both directions
- Smart Tool measurement range to saturation
- Confidence intervals for static sensitivities

Experiment 2 – Calibration of Bending Crosstalk

Objective

The purpose of this experiment is to characterize the cross-sensitivity of transverse bending loads on the tool holder.

Equipment List

- C-channel Instron Fixture
- Instron servo-hydraulic 55s
- Load cell
- 2 point contact
- Rapid-prototyped angular alignment block
- Level
- Smart Tool v.10
- Laptop with Bluetooth capability and MATLAB

Experimental Protocol

Load Cell Calibration

The calibration file must be loaded for the appropriate load cell. Use the drop down menu bar for the Instron software application to load the appropriate calibration file.

Smart Tool Calibration of Bending Sensitivity

4. Attach the U-channel fixture to the Instron machine
5. Attach the Smart Tool to the fixture to calibrate the radial bending cross-sensitivity
 - a. Tighten all bolts to 25 in-lbf
 - b. Record the distance from the gages to the point of contact
 - i. Measure this distance 10 times to assign a statistical confidence to the lever arm measurement.
 - c. Load the tool from 0 lbf to 200 lbf in roughly 10 lbf increments. Loading should be applied in the tangential direction, perpendicular to the radial bridge. In other words, to calibrate bending crosstalk in the radial direction, the load is applied with the radial bridge on the neutral axis.
 - d. Unload the tool from 200 lbf to 0 lbf in roughly 10 lbf increments
 - e. Repeat this loading four more times for a total of five loading cycles
 - i. Record all measurements from the Instron load cell
 - ii. Record all radial strain measurements from the Smart Tool using the sth10live.m file with a moving window of 20 seconds.
6. Repeat step 5 to calibrate the tangential bending crosstalk.

Characterized Parameters

- Load cell sensitivity and measurement confidence
- Smart Tool static cross-sensitivity to bending in both directions
- Theoretical circumferential misalignment of the strain gages

Experiment 3 – Calibration of Torsional Crosstalk

Objective

The purpose of this experiment is to characterize the torsional cross-sensitivity of the sensor.

Equipment List

- Smart Tool v.10
- Static calibration test fixture
- Weights necessary to produce full-scale torsional strains

Experimental Protocol

Experimental Setup

- Fix the sensor in the static calibration test fixture, set up to apply torsional loads.
- Using the rapid-prototyped alignment block, orient the toolholder to that the radial bridge is facing up. This is best achieved by using a level.
- Power on the Smart Tool and wait 5 minutes before starting the experiment.

Performing the Experiment

1. Load the sensor with increasing torque over the range of 0 to 115 in-lbf. Approximately ten points should be used over the full scale range.
 - a. Record all measurements using the “sth10_live.m” file and a moving average of 20 seconds.
2. Un-load the sensor in decreasing order of applied weights.
 - a. Record all measurements using the “sth10_live.m” file and a moving average of 20 seconds.
3. Repeat steps 1-2 for a total of 5 times.
4. Repeat steps 1-3 with the sensor rotated 90 degrees such that the tangential bridge is facing up.

Characterized Parameters

- Torsional cross sensitivity on the radial bridge
- Torsional cross-sensitivity on the tangential bridge
- Theoretical planar misalignment of the Wheatstone bridge

Experiment 4 – Calibration of Axial Sensitivity

Objective

The purpose of this experiment is to characterize the axial sensitivity of the sensor.

Equipment List

- Smart Tool v.10
- Rapid-prototyped block to hold the sensor at the retainment groove
- Cutting insert with wire through the screw holes to make a loop for the weight stand
- Weight stand and static calibration weights, 0 lbf to 30 lbf

Experimental Protocol

Experimental Setup

- Fix the sensor in the rapid prototyped block and secure it in the bench vise
- Make a wire loop so that it is possible to suspend axial loads from the tool holder
- Power on the Smart Tool and wait 5 minutes before starting the experiment

Performing the Experiment

1. Load the sensor with increasing axial load over the range of 0 to 30 lbf. Approximately ten points should be used over this range.
 - a. Record all measurements using the “sth10_live.m” file and a moving average of 20 seconds.
2. Un-load the sensor in decreasing order of applied weights.
 - a. Record all measurements using the “sth10_live.m” file and a moving average of 20 seconds.
3. Repeat steps 1-2 to obtain five total data sets

Characterized Parameters

- Axial sensitivity on the radial bridge
- Axial sensitivity on the tangential bridge

Experiment 5 – Characterization of DC Drift

Objective

The purpose of this experiment is to characterize the DC drift of the sensor.

Equipment List

- Smart Tool v.10
- CNC Machine
- Stop watch

Experimental Protocol

Experimental Setup

- Fix the tool in the CNC spindle
- Power on the Smart Tool and wait 5 minutes before starting the experiment

Performing the Experiment

1. Power on the tool with the bridge selection set to the radial bridge
 - a. Record measurements in approximately one minute intervals for 15 minutes
2. Repeat step 1 for a total of 5 records on the radial bridge
3. Power on the tool with the bridge selection set to the tangential bridge
 - a. Record measurements in approximately one minute intervals for 15 minutes
4. Repeat step 1 for a total of 5 records on the tangential bridge

Characterized Parameters

- DC drift with constant environmental conditions

Raw Calibration Data

Bending Sensitivity Calibration Raw Data

Table A.1 – Calibration of radial bending sensitivity

Tangential Run 1		Tangential Run 2		Tangential Run 3		Tangential Run 4		Tangential Run 5	
Load (N)	Strain (bits)	Load (N)	Strain (bits)	Load (N)	Strain (bits)	Load (N)	Strain (bits)	Load (N)	Strain (bits)
0.00	32576	0.00	32772	0.00	32692	0.00	33082	0.00	33069
71.47	34285	77.72	34721	60.78	34254	72.42	34798	56.81	34347
146.22	36102	145.47	36221	152.25	36260	148.31	36550	144.79	36451
215.86	37760	206.74	37733	213.86	37899	217.20	38342	223.17	38251
300.82	39750	300.73	39744	295.20	39774	290.03	39881	283.04	39575
374.84	41459	369.88	41567	350.07	40964	361.94	41526	366.18	41485
451.05	43264	424.88	42938	439.82	43233	442.87	43482	435.39	43290
507.93	44595	490.66	44181	501.24	44758	476.51	44430	507.84	45058
580.13	46272	586.04	46628	575.27	46376	575.94	46745	572.73	46595
660.71	48115	654.09	48142	645.28	47963	645.50	48429	646.72	48269
705.20	49146	720.36	49732	726.71	49870	715.31	49950	718.66	50136
646.86	47699	650.55	48234	666.03	48421	649.44	48425	646.72	48295
575.39	45997	582.97	46681	584.96	46658	581.34	46627	561.44	46221
512.67	44512	499.69	44587	504.29	44743	501.15	44991	503.82	44905
427.35	42496	447.54	43323	424.52	42748	432.18	43258	420.10	42855
353.69	40762	363.47	41553	360.84	41387	343.50	41175	361.28	41586
285.51	39149	280.34	39406	278.02	39254	276.86	39688	276.34	39658
213.67	37453	211.07	37811	215.13	38007	212.27	38037	229.63	38503
112.67	35091	159.05	36510	153.42	36204	142.30	36448	152.21	36596
0.00	32461	65.50	34377	71.13	34432	61.33	34423	66.61	34417
		0.00	32663	0.00	32845	0.00	32988	0.00	32909

Table A.2 – Calibration of tangential bending sensitivity

Radial Run 1		Radial Run 2		Radial Run 3		Radial Run 4		Radial Run 5	
Load (N)	Strain (bits)	Load (N)	Strain (bits)	Load (N)	Strain (bits)	Load (N)	Strain (bits)	Load (N)	Strain (bits)
0.00	32653	0.00	32263	0.00	33696	0.00	33456	0.00	31782
76.94	34502	75.19	34098	76.38	35441	73.35	35055	80.53	33814
153.51	36327	166.04	36318	147.30	37342	157.77	37138	123.27	34752
220.97	37927	213.32	37551	202.00	38401	221.46	38961	203.34	36439
311.40	40071	294.38	39231	282.78	40411	276.34	40025	280.98	38460
359.89	41197	359.81	40863	370.05	42513	337.81	41445	356.72	40245
437.19	43009	436.32	42570	420.62	43557	433.39	43636	418.88	41810
513.04	44781	488.10	43840	502.47	45651	501.73	45111	510.55	43810
593.62	46657	578.50	46221	566.26	47057	581.55	47173	570.77	45244
654.15	48065	655.63	47791	638.20	48568	657.14	48649	646.35	46849
719.78	49582	719.94	49231	710.79	50372	718.00	50284	714.51	48558
657.07	48046	644.40	47604	656.23	49119	649.49	48622	647.12	46875
579.77	46190	576.26	45993	576.07	47306	568.79	46840	589.72	45651
515.23	44660	483.11	43571	507.74	45706	516.01	45719	485.70	43144
444.85	42984	444.42	42744	426.45	43791	432.52	43702	422.39	41606
377.39	41364	363.42	40856	358.75	41997	354.67	41780	355.29	40148
295.72	39425	293.51	39229	296.37	40521	283.43	39992	293.55	38617
214.77	37505	222.55	37593	203.13	38598	234.18	38937	205.77	36750
156.06	36104	146.87	35750	136.41	36979	149.08	36873	138.75	35263
77.30	34235	86.18	34415	82.21	35665	76.20	35310	79.04	33663
0	32417.7	0.00	32391	0.00	33837	0.00	33421	0.00	31815

Bending Cross-Sensitivity Calibration Raw Data

Radial bending cross-sensitivity:

Load is applied in the tangential direction with the radial bridge on the neutral axis.

Table A.3 – Calibration of radial bending cross-sensitivity

Radial Run 1		Radial Run 2		Radial Run 3		Radial Run 4		Radial Run 5	
Load (N)	Strain (bits)	Load (N)	Strain (bits)	Load (N)	Strain (bits)	Load (N)	Strain (bits)	Load (N)	Strain (bits)
0.00	32801	0.00	32918	0.00	34093	0.00	31992	0.00	32663
59.04	32808	88.27	32928	65.53	34101	58.65	31999	60.64	32670
129.07	32816	154.10	32937	141.48	34110	155.73	32011	157.81	32682
205.37	32826	239.31	32947	217.90	34118	214.10	32018	224.46	32689
289.39	32835	293.54	32953	283.32	34127	291.96	32028	285.00	32697
359.42	32845	362.05	32961	365.31	34137	367.45	32037	351.14	32705
429.62	32853	444.39	32971	434.45	34145	429.49	32044	428.22	32713
513.36	32862	505.35	32978	523.46	34154	505.65	32053	507.82	32725
574.48	32869	575.81	32987	563.23	34161	571.65	32062	580.29	32731
653.16	32879	638.56	32995	648.07	34171	659.71	32072	654.75	32741
736.27	32889	712.26	33003	716.37	34179	710.91	32078	714.49	32749
659.58	32880	647.63	32996	660.67	34172	661.01	32071	648.42	32740
571.12	32870	582.62	32989	581.38	34163	586.41	32062	574.32	32730
526.20	32863	513.78	32980	512.07	34154	518.75	32055	497.84	32722
441.72	32853	425.45	32969	430.89	34144	434.87	32045	440.37	32715
362.32	32844	355.27	32962	369.13	34137	374.46	32037	363.60	32706
277.35	32834	279.16	32952	303.25	34129	287.40	32027	284.60	32697
233.03	32829	203.51	32943	209.08	34118	214.42	32018	212.01	32688
142.88	32818	146.92	32936	134.98	34110	139.75	32010	143.72	32680
70.86	32810	69.02	32927	72.70	34101	76.55	32002	65.05	32671
0	32801	0.00	32919	0.00	34092	0.00	31993	0.00	32663

Tangential bending cross-sensitivity:

Load is applied in the radial direction with the tangential bridge on the neutral axis.

Table A.4 – Calibration of tangential bending cross-sensitivity

Tangential Run 1		Tangential Run 2		Tangential Run 3		Tangential Run 4		Tangential Run 5	
Load (N)	Strain (bits)	Load (N)	Strain (bits)	Load (N)	Strain (bits)	Load (N)	Strain (bits)	Load (N)	Strain (bits)
0.00	32925	0.00	32836	0.00	32723	0.00	32604	0.00	32244
94.13	32991	80.74	32883	69.25	32767	69.47	32640	68.06	32289
157.53	33027	138.88	32923	132.96	32810	141.87	32691	141.08	32332
215.20	33062	220.52	32971	216.81	32855	194.73	32725	220.47	32385
303.89	33116	300.30	33025	280.14	32904	285.64	32780	305.54	32435
356.48	33153	358.14	33056	345.36	32944	342.92	32817	351.84	32465
421.33	33191	434.18	33106	435.97	32996	435.22	32873	450.71	32527
499.44	33240	477.67	33134	498.36	33037	511.40	32925	505.35	32559
569.93	33286	571.29	33199	579.78	33090	569.03	32960	569.24	32608
646.75	33335	650.00	33241	659.70	33138	654.92	33014	654.49	32661
719.31	33381	721.10	33286	719.54	33176	724.28	33060	723.92	32700
656.26	33346	664.62	33254	641.65	33127	648.12	33009	642.58	32652
568.95	33289	579.41	33196	579.31	33089	593.56	32978	577.27	32612
514.00	33256	503.98	33149	512.82	33045	511.72	32923	489.66	32551
428.94	33192	437.22	33110	436.19	33000	431.69	32874	406.39	32498
365.12	33157	347.07	33053	359.59	32950	367.01	32832	363.04	32472
306.39	33120	283.32	33015	287.30	32908	294.50	32787	277.17	32422
217.36	33064	220.97	32973	205.85	32851	232.84	32747	222.35	32385
149.02	33025	141.67	32926	142.66	32815	130.10	32683	150.03	32338
65.06	32967	48.06	32865	71.59	32769	79.29	32648	82.18	32298
0	32925	0.00	32837	0.00	32724	0.00	32602	0.00	32241

Torsional Cross-Sensitivity Calibration Raw Data

Table A.5 – Calibration of torsional cross sensitivity

Applied Torque (N-m)	Radial Strain (bits)		Tangential Strain (bits)		
	Run 1	Run 2	Run 1	Run 2	Run 3
0.00	30682	30726	32205	32166	32115
1.25	30683	30728	32207	32157	32104
7.48	30670	30713	32224	32163	32108
7.96	30667	30715	32226	32165	32111
8.90	30657	30711	32227	32166	32115
10.03	30650	30708	32229	32172	32119
11.16	30642	30702	32231	32174	32125
12.29	30636	30698	32233	32178	32127
13.42	30633	30696	32235	32180	32128
14.55	30631	30696	32235	32178	32127
15.68	30629	30696	32235	32178	32127
14.55	30621	30691	32229	32172	32123
13.42	30612	30683	32220	32166	32117
12.29	30604	30677	32203	32155	32111
11.16	30606	30681	32197	32153	32106
10.03	30612	30689	32193	32149	32102
8.90	30619	30698	32190	32144	32100
7.96	30627	30706	32186	32142	32098
7.48	30634	30713	32182	32140	32096
1.25	30689	30772	32186	32138	32098
0.00	30693	30777	32197	32151	32108

Axial Sensitivity Calibration Raw Data

Table A.6 – Calibration of axial sensitivity

Applied Load (N)	Tangential Strain (bits)				
	Run 1	Run 2	Run 3	Run 4	Run 5
0.00	32176	32895	32059	33721	33122
9.83	32186	32904	32069	33730	33133
58.94	32241	32958	32122	33785	33185
62.65	32246	32962	32127	33788	33190
70.08	32255	32972	32135	33796	33199
78.97	32265	32983	32146	33807	33208
87.87	32272	32991	32155	33816	33219
96.77	32283	33002	32164	33826	33229
105.66	32293	33012	32175	33836	33237
114.56	32303	33022	32184	33846	33247
105.66	32294	33012	32175	33837	33239
96.77	32284	33000	32166	33827	33228
87.87	32273	32992	32157	33817	33217
78.97	32264	32981	32145	33807	33208
70.08	32254	32970	32135	33797	33198
62.65	32245	32963	32128	33787	33189
58.94	32243	32959	32124	33784	33186
9.83	32187	32907	32069	33729	33132
0.00	32177	32895	32059	33720	33122

Drift Investigation Raw Data

Table A.7 - Table of raw data for drift investigation

Time (min)	Radial Strain (bits)	Tangential Strain (bits)	Time (min)	Radial Strain (bits)	Time (min)	Radial Strain (bits)	Time (min)	Tangential Strain (bits)
0	30893	30848	0	32403	0	32589	0	32749
1	30886	30976	1	32378	1	32582	1	32774
2	30874	31078	2	32320	2	32563	2	32800
3	30867	31130	3	32250	3	32538	3	32813
4	30848	31155	4	32179	4	32518	4	32819
5	30835	31168	5	32141	5	32499	5	32826
6	30822	31174	6	32083	6	32480	6	32819
7	30810	31174	7	32038	7	32461	7	32819
8	30797	31174	8	31987	8	32448	8	32819
9	30784	31174	9	31942	9	32429	9	32819
10	30771	31168	10	31898	10	32416	10	32813
11	30758	31168	11	31917	11	32397	11	32813
12	30752	31168			12	32384	12	32813
13	30739	31162			13	32371	13	32813
14	30733	31162			14	32358	14	32806
15	30726	31162			15	32346	15	32806
					16	32333	16	32800
					17	32320	17	32800
					18	32307	18	32800
					19	32301	19	32800
					20	32288	20	32794
					21	32275	21	32800
					22	32262	23	32794
					23	32256	24	32794
					37	32134	25	32794
							26	32794
							27	32787
							28	32787
							29	32787
							30	32787

APPENDIX B

PROTOCOL FOR CONSTRUCTION OF SMART TOOL V.10

Overview

The performance of the sensor is dependent on how well design tolerances are maintained during the assembly process. Factors such as strain gage alignment, surface preparation, and adhesive selection are critical in the construction process. The following sections detail my procedure for sensor construction.

Gage Alignment

It is critical that the strain gages be aligned with the cutting tooth. The orientation of the cutting insert holder is predefined by a set screw located on the tool holder body. The gages must be aligned with the radial and tangential components of force acting on the cutting tooth. The following procedure outlines our methodology for achieving sufficient alignment of the gage mounting locations.

Finding the gage mounting locations

1. Mount the tool holder in the vice attached to the table of the CNC machine
2. Secure a pencil in the CNC spindle via the chuck-attachment tool holder
3. Align the pencil tip with the cutting tooth
 - a. Because of the rake angle of the tooth, offset the pencil tip 0.0625" from the tool end.
This centers the gage location for cuts with an axial depth of 0.125 inches.
4. Using the handwheel, move the pencil to the tool holder body and scribe a line along the surface
5. Rapid prototype a square alignment block
 - a. Use a side length equal to the diameter of the taper flange.
 - b. Create a center hole that forms an interference fit with the tool holder body
 - c. Extrude a small key to orient with the scribed line
6. Fix the alignment block to the tool holder body; align the key with the scribed pencil line
7. Mill a flat for the radial gage using the alignment block to orient the tool holder
8. Mill a flat for the tangential gage

Gage Mounting Procedure

The details of mounting strain gages are non-trivial. A rigid, secure bond between the strain gage and the tool holder is necessary for the gages to function properly with no creep and a linear response to strain. Our methodology for gage mounting is presented here:

1. Sand the flats to achieve the necessary surface roughness for the adhesive to bond properly
2. Clean the flats with acetone
3. Use Loctite 380 to black out the gages and the mounting locations
 - a. Semi-conductor strain gages are highly sensitive to light
 - b. Apply only a thin layer; while the adhesive is fairly rigid, too much adhesive will add unnecessary compliance and can introduce error due to gage creep
 - c. Let cure for 24 hours
4. Bond the strain gages to the flats using Loctite 401
 - a. Apply a thin coat evenly to the flat; spread with a graphite pencil if necessary
 - b. Use tweezers to place the gage; use raised edge to achieve planar alignment
 - c. Clamp the gage using a rubber pad and a quick clamp
 - d. Let cure for 24 hours
 - e. Rotate the tool holder and repeat steps (a)-(d) for the other gage

Charging Circuit

The following steps were performed to install the charging circuit beneath the retention bolt:

1. Drill a 1/4" hole in the tool holder body using a carbide steel drill bit
2. Solder the charging jack to the charging circuit board
3. Coat the charging circuit board in a strong epoxy to protect components
4. Pass the power cables through the 1/4" hole in the tool holder body
5. Ensure that the charging jack is positioned such that it does not interfere with the retaining bolt
6. Secure the charging jack in place with epoxy

Wiring Protocol

1. Solder leadwires to the appropriate bridge terminals for each strain gage
2. Crimp terminal connectors to each gage and to the battery wires
3. Solder the battery in parallel with the charging circuit
4. Attach terminal connectors to appropriate pin headers on data transmission board

Component Testing

After the construction of the sensor is complete, individual components are tested to ensure that the assembly was successful. The following outlines our testing methodology:

1. Measure the resistance across the bridge in both directions (for both strain gages) to ensure that it is nominally 500 ohms
2. Inspect the solder points on the strain gage terminals to ensure that nothing is accidentally grounded to the tool holder body
3. Power on each data transmission board to verify that the board boots properly
4. Pair the data transmission boards with the host computer
 - a. Set the baud rate to 115200
 - b. Set the number of bits to 8
5. Ensure Bluetooth connectivity through Matlab
6. Verify that the DAC can balance the Wheatstone bridge and no saturation occurs

APPENDIX C

DERIVATION OF CROSSTALK THEORY DUE TO BRIDGE MISALIGNMENT

Forces Acting on a Single Cutting Tooth:

Tangential Force:	$F_t := 100\text{-lbf}$	The ratio of these loads is changed to obtain the curves for each loadcase
Radial Force:	$F_r := 20\text{-lbf}$	

Small Angle Misalignment Arrays:

	$j := 24$	
Range Variable:	$i := 0..j$	
Circumferential Misalignment Angle:	$\theta_i := \frac{i}{4} \cdot \text{deg}$	4 steps per degree, 0.6 degrees
Planar Misalignment Angle:	$\phi_i := \frac{i}{4} \cdot \text{deg}$	4 steps per degree, 0.6 degrees

Geometric Properties of the Tool Holder:

Metric Units

Outer Radius:	$R := \frac{1.75}{2} \cdot \text{in}$	R = 0.022m
Inner Radius:	$r := \frac{0.75}{2} \cdot \text{in}$	$r = 9.525 \times 10^{-3} \text{ m}$
Cross Sectional Area:	$A_c := \pi \cdot (R^2 - r^2)$	$A_c = 1.267 \times 10^{-3} \cdot \text{m}^2$
Area Moment of Inertia:	$I := \frac{\pi}{4} \cdot (R^4 - r^4)$	$I = 1.852 \times 10^{-7} \cdot \text{m}^4$
Polar Moment of Inertia:	$J := \frac{\pi}{2} \cdot (R^4 - r^4)$	$J = 3.703 \times 10^{-7} \cdot \text{m}^4$
Thickness at Shear:	$t_{1_i} := 2 \cdot (R - r) \cdot \cos(\theta_i)$	
	$t_{2_i} := 2 \cdot R \cdot \sin(\theta_i)$	

Length to Gages:

$$l := 5.1 \cdot \text{in}$$

$$l = 0.13 \text{m}$$

Equivalent System of Forces and Couples Acting at the Center of the Cross-Section

Twisting Couple: $T_c := F_t \cdot r$

Bending Couples: $M_y := F_r \cdot l$

$$M_z := F_t \cdot l$$

First Moment of Area Calculations:

Area of Interest:
$$A_{m_1} := 2 \cdot \int_0^{R \cdot \cos(\theta_i)} \sqrt{R^2 - x^2} dx - 2 \cdot \int_0^{r \cdot \cos(\theta_i)} \sqrt{r^2 - x^2} dx$$

Functions Describing the Geometry for the First Moment of Area:

Upper Bounding Curves: Outer Radius: $F(x) := \sqrt{R^2 - x^2}$

Inner Radius: $f(x) := \sqrt{r^2 - x^2}$

Lower Bounding Curves: Outer Radius: $G_{1_i} := R \cdot \sin(\theta_i)$

$$G_{2_i} := R \cdot \cos(\theta_i)$$

Inner Radius: $g_{1_i} := R \cdot \sin(\theta_i)$

$$g_{2_i} := R \cdot \cos(\theta_i)$$

Limits of Integration: $A_{1_i} := -R \cdot \cos(\theta_i)$ $a_{1_i} := -r \cdot \cos(\theta_i)$ $B_{1_i} := R \cdot \cos(\theta_i)$ $b_{1_i} := r \cdot \cos(\theta_i)$

$$A_{2_i} := -R \cdot \sin(\theta_i) \qquad B_{2_i} := R \cdot \sin(\theta_i)$$

First Moments of Area:

$$Q_{1_i} := \frac{1}{2} \cdot \int_{A_{1_i}}^{B_{1_i}} (F(x) + G_{1_i}) \cdot (F(x) - G_{1_i}) dx - \frac{1}{2} \cdot \int_{a_{1_i}}^{b_{1_i}} (f(x) + g_{1_i}) \cdot (f(x) - g_{1_i}) dx$$

$$Q_{2_i} := \frac{1}{2} \cdot \int_{A_{2_i}}^{B_{2_i}} (F(x) + G_{2_i}) \cdot (F(x) - G_{2_i}) dx$$

Normal Stress Calculations for each Gage:

Normal Stresses Produced by Bending Couple M_y :

$$\sigma_{A_{t_i}} := \frac{-M_y \cdot R \cdot \sin(\theta_i)}{I}$$

$$\sigma_{B_{a_i}} := \frac{-M_y \cdot R \cdot \cos(\theta_i)}{I}$$

Normal Stresses Produced by Bending Couple M_z :

$$\sigma_{A_{a_i}} := \frac{-M_z \cdot R \cdot \cos(\theta_i)}{I}$$

$$\sigma_{B_{t_i}} := \frac{M_z \cdot R \cdot \sin(\theta_i)}{I}$$

Shear Stress Calculations for each Gage:

Shear Stresses Produced by Twisting Couple T:

$$\tau_{\text{twist}} := \frac{T_c \cdot R}{J}$$

Shear Stresses Produced by Shear Force F_t :

$$i := 1..j \quad \tau_{A1_i} := \frac{F_t \cdot Q_{2_i}}{I \cdot t_{2_i}}$$

(Skip first index to avoid division by zero)

$$i := 0..j \quad \tau_{B1_i} := \frac{F_t \cdot Q_{1_i}}{I \cdot t_{1_i}}$$

Shear Stresses Produced by Shear Force F_r :

$$i := 0..j \quad \tau_{A2_i} := \frac{F_r \cdot Q_{1_i}}{I \cdot t_{1_i}}$$

$$i := 1..j \quad \tau_{B2_i} := \frac{F_r \cdot Q_{2_i}}{I \cdot t_{2_i}} \quad (\text{Skip first index to avoid division by zero})$$

(reset) $i := 0..j$

Stresses for Perfectly Aligned Gages:

Gages at Location A:

Axial Stress: $\sigma_{Aa_0} = -1.003 \times 10^3 \cdot \text{psi}$

Transverse Stress: $\sigma_{At_0} = 0 \cdot \text{psi}$

Shear Stress: $\tau_{A_i} := \tau_{\text{twist}} + \tau_{A1_i} + \tau_{A2_i} \quad \tau_{A_0} = 3.818 \times 10^5 \text{ Pa}$

Gages at Location B:

Axial Stress: $\sigma_{Ba_0} = -200.627 \text{ psi}$

Transverse Stress: $\sigma_{Bt_0} = 0 \cdot \text{psi}$

Shear Stress: $\tau_{B_i} := \tau_{\text{twist}} + \tau_{B1_i} + \tau_{B2_i} \quad \tau_{B_0} = 8.92 \times 10^5 \text{ Pa}$

Stress-Strain Relations:

Material Properties:	<i>Material:</i>	Steel
	<i>Young's Modulus:</i>	$E := 30 \cdot 10^6 \text{ psi}$
	<i>Shear Modulus:</i>	$G := 11.2 \cdot 10^6 \cdot \text{psi}$
	<i>Poisson's Ration:</i>	$\nu := .288$

Calculated Strain at Each Gage Location (for perfectly aligned gages):

Gages at Location A:

$$\text{Axial Strain: } \varepsilon_{Aa_i} := \frac{1}{E} \cdot (\sigma_{Aa_i} - \nu \cdot \sigma_{At_i}) \quad \varepsilon_{Aa_0} = -3.344 \times 10^{-5}$$

$$\text{Transverse Strain: } \varepsilon_{At_i} := \frac{1}{E} \cdot (\sigma_{At_i} - \nu \cdot \sigma_{Aa_i}) \quad \varepsilon_{At_0} = 9.63 \times 10^{-6}$$

$$\text{Shear Strain: } \gamma_{A_i} := \frac{\tau_{A_i}}{G} \quad \gamma_{A_0} = 4.945 \times 10^{-6}$$

Gages at Location B:

$$\text{Axial Strain: } \varepsilon_{Ba_i} := \frac{1}{E} \cdot (\sigma_{Ba_i} - \nu \cdot \sigma_{Bt_i}) \quad \varepsilon_{Ba_0} = -6.688 \times 10^{-6}$$

$$\text{Transverse Strain: } \varepsilon_{Bt_i} := \frac{1}{E} \cdot (\sigma_{Bt_i} - \nu \cdot \sigma_{Ba_i}) \quad \varepsilon_{Bt_0} = 1.926 \times 10^{-6}$$

$$\text{Shear Strain: } \gamma_{B_i} := \frac{\tau_{B_i}}{G} \quad \gamma_{B_0} = 1.155 \times 10^{-5}$$

Calculated Change in Wheatstone Bridge Output (for perfectly aligned gages):

Governing Equation:

$$\delta E_o = \frac{G_F \cdot E_i}{4} \cdot (\varepsilon_1 - \varepsilon_2 + \varepsilon_3 - \varepsilon_4) \cdot (1 - n)$$

We make the following definitions:

$$\text{Gage Factor: } G_F := 140 \quad (\text{Semiconductor Strain Gages})$$

$$\text{Supply Voltage: } E_i := 3.3 \cdot \text{V}$$

$$\text{Nonlinearity Factor: } n = \frac{1}{1 + \frac{2}{G_F \cdot (\varepsilon_1 + \varepsilon_2 + \varepsilon_3 + \varepsilon_4)}} \quad n := 0$$

The resulting output error is thus

$$\text{Error} = \frac{|\delta E_{\text{aligned}} - \delta E_{\text{misaligned}}|}{\delta E_{\text{aligned}}} \cdot 100$$

Gage Perturbation Analysis:

Sensitivity of Bridge B to Circumferential Misalignment (moving one gage location):

$$\delta E_{o_i} := \frac{G_F \cdot E_i}{4} \cdot (-2 \varepsilon_{Ba_i}) \cdot (1 - n)$$

$$\text{Err}_{B_i} := \frac{|\delta E_{o_0} - \delta E_{o_i}|}{\delta E_{o_0}} \cdot 100$$

Sensitivity of Bridge A to Circumferential Misalignment (moving one gage location):

$$\delta E_{o_i} := \frac{G_F \cdot E_i}{4} \cdot (-2 \varepsilon_{Aa_i}) \cdot (1 - n)$$

$$\text{Err}_{A_i} := \frac{|\delta E_{o_0} - \delta E_{o_i}|}{\delta E_{o_0}} \cdot 100$$

Sensitivity of Bridge B to Planar Misalignment (at one gage location):

$$\varepsilon_{Ba2_i} := \frac{\varepsilon_{Ba_0} + \varepsilon_{Bt_0}}{2} + \frac{\varepsilon_{Ba_0} - \varepsilon_{Bt_0}}{2} \cdot \cos(2 \cdot \phi_i) + \gamma_{B_0} \cdot \sin(2 \cdot \phi_i)$$

$$\delta E_{o_i} := \frac{G_F \cdot E_i}{4} \cdot (-2 \varepsilon_{Ba2_i}) \cdot (1 - n)$$

$$\text{Err}_{B2_i} := \frac{|\delta E_{o_0} - \delta E_{o_i}|}{\delta E_{o_0}} \cdot 100$$

Sensitivity of Bridge A to Planar Misalignment (at one gage location):

$$\varepsilon_{Aa2_i} := \frac{\varepsilon_{Aa_0} + \varepsilon_{At_0}}{2} + \frac{\varepsilon_{Aa_0} - \varepsilon_{At_0}}{2} \cdot \cos(2 \cdot \phi_i) + \gamma_{A_0} \cdot \sin(2 \cdot \phi_i)$$

$$\delta E_{o_i} := \frac{G_F \cdot E_i}{4} \cdot (-2 \varepsilon_{Aa2_i}) \cdot (1 - n)$$

$$\text{Err}_{A2_i} := \frac{|\delta E_{o_0} - \delta E_{o_i}|}{\delta E_{o_0}} \cdot 100$$

APPENDIX D

DATA REDUCTION PROTOCOL FOR EXPERIMENTAL VALIDATION

Presented here is an overview of the data reduction procedure used for the net force comparison. From a macro-perspective, the net force profiles are aligned by manipulating the data as shown in Figure D.1.

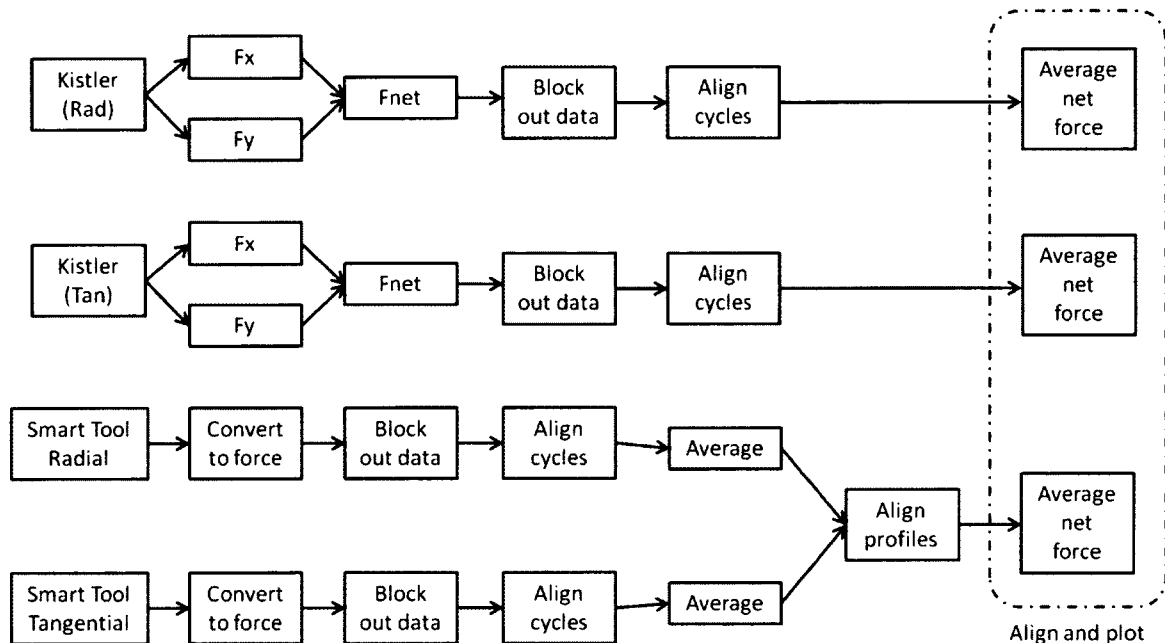


Figure D.1 – Block diagram of data reduction for net force comparison

At 600 RPM, net force profiles were compared for 12 unique cutting conditions ($\frac{1}{4}$, $\frac{1}{2}$, $\frac{3}{4}$ immersion, $h_{avg} = 0.001, 0.002, 0.003, 0.004$ in). 36 unique conditions were compared at higher spindle speeds (3000, 3600, 4000 RPM, $\frac{1}{4}$, $\frac{1}{2}$, $\frac{3}{4}$ immersion, $h_{avg} = 0.001, 0.002, 0.003, 0.004$ in).

Some of the Kistler data from the cutting test was corrupted due to a problem with the data acquisition system. This bad data is highlighted in Figure D.2.

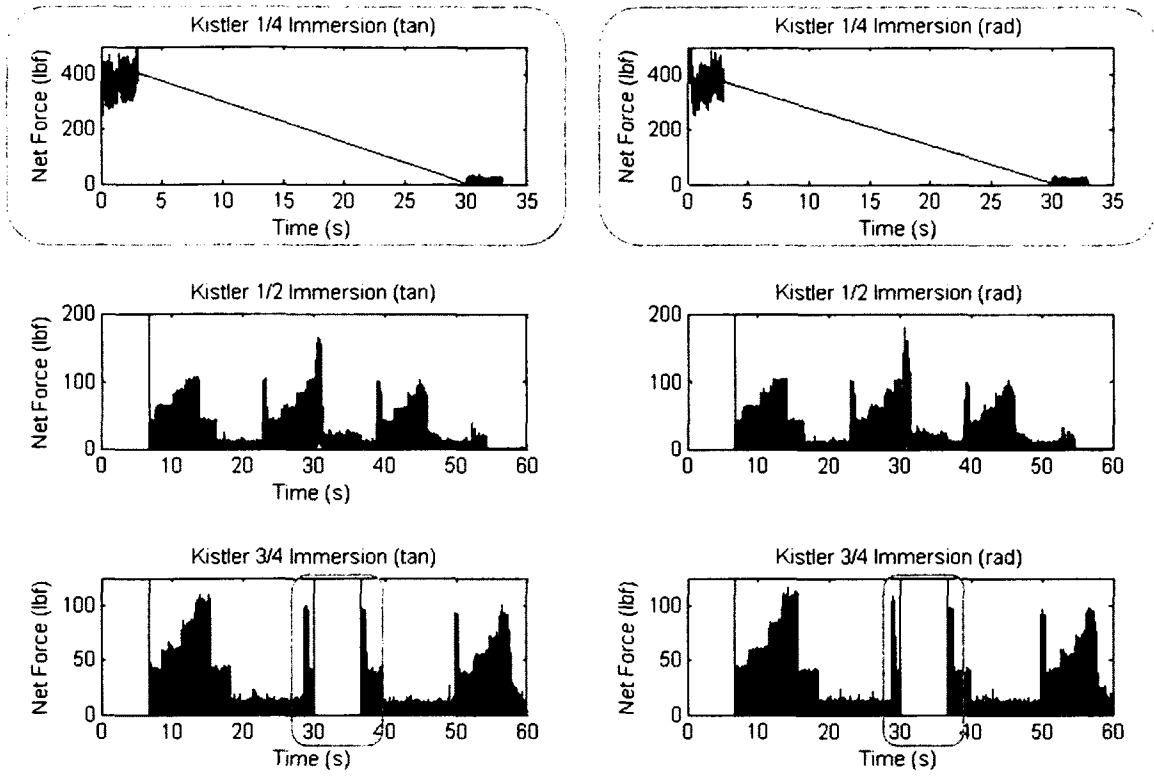


Figure D.2 - Some bad Kistler data in the experiment

For each feedrate, sections of good data were blocked out using the cursor tool.

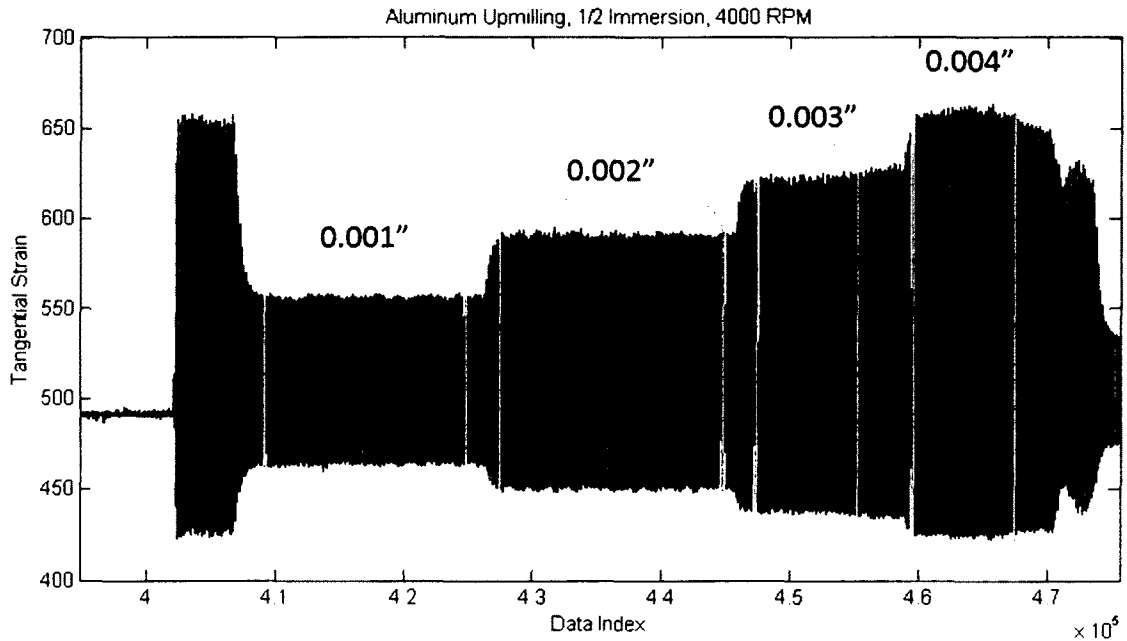


Figure D.3 - Blocking out sections of data to average

Table D.1 shows the starting and stopping data indices for each cutting condition.

Table D.1 – Table of indices used to block out data

Immerision	Chip Load	Smart Tool Tangential		Smart Tool Radial		Kistler Tangential		Kistler Radial		
		Start	Stop	Start	Stop	Start	Stop	Start	Stop	
		3,000 RPM Aluminum Upmilling	1/4	h1	3.22E+04	5.03E+04	4.31E+04	6.06E+04		
h2	5.13E+04			6.98E+04	6.20E+04	8.03E+04				
h3	7.10E+04			8.30E+04	8.15E+04	9.35E+04				
h4	8.39E+04			9.75E+04	9.44E+04	1.08E+05				
1/2	h1		5.66E+04	8.12E+04	6.07E+04	8.53E+04	2.51E+05	2.92E+05	2.54E+05	2.93E+05
	h2		8.23E+04	1.07E+05	8.75E+04	1.12E+05	1.39E+05	1.79E+05	1.43E+05	1.85E+05
	h3		1.08E+05	1.25E+05	1.13E+05	1.29E+05	1.86E+05	2.11E+05	1.89E+05	2.16E+05
	h4		1.29E+05	1.44E+05	1.30E+05	1.48E+05	2.16E+05	2.46E+05	2.20E+05	2.50E+05
3/4	h1		5.92E+04	8.49E+04	8.78E+04	1.13E+05	1.55E+05	2.02E+05	2.83E+05	3.30E+05
	h2		9.15E+04	1.15E+05	1.16E+05	1.42E+05	2.10E+05	2.36E+05	1.58E+05	2.04E+05
	h3		1.22E+05	1.34E+05	1.46E+05	1.62E+05	2.42E+05	2.76E+05	2.11E+05	2.39E+05
	h4		1.39E+05	1.56E+05	1.66E+05	1.84E+05	2.81E+05	3.27E+05	2.44E+05	2.78E+05

Immerision	Chip Load	Smart Tool Tangential		Smart Tool Radial		Kistler Tangential		Kistler Radial		
		Start	Stop	Start	Stop	Start	Stop	Start	Stop	
		3,600 RPM Aluminum Upmilling	1/4	h1	2.26E+05	2.41E+05	2.37E+05	2.51E+05		
h2	2.42E+05			2.57E+05	2.52E+05	2.67E+05				
h3	2.58E+05			2.68E+05	2.69E+05	2.78E+05				
h4	2.69E+05			2.80E+05	2.80E+05	2.90E+05				
1/2	h1		2.47E+05	2.62E+05	2.50E+05	2.67E+05	4.60E+05	4.95E+05	4.26E+05	4.60E+05
	h2		2.65E+05	2.84E+05	2.71E+05	2.89E+05	4.98E+05	5.19E+05	4.65E+05	4.99E+05
	h3		2.87E+05	2.98E+05	2.94E+05	3.04E+05	5.24E+05	5.39E+05	5.03E+05	5.24E+05
	h4		3.01E+05	3.15E+05	3.07E+05	3.20E+05	4.21E+05	4.56E+05	5.28E+05	5.39E+05
3/4	h1		3.03E+05	3.22E+05	3.28E+05	3.49E+05	6.76E+05	7.14E+05	6.77E+05	7.17E+05
	h2		3.25E+05	3.46E+05	3.52E+05	3.74E+05				
	h3		3.50E+05	3.63E+05	3.77E+05	3.89E+05				
	h4		3.66E+05	3.81E+05	3.92E+05	4.08E+05	6.61E+05	6.72E+05	6.61E+05	6.74E+05

Immerision	Chip Load	Smart Tool Tangential		Smart Tool Radial		Kistler Tangential		Kistler Radial		
		Start	Stop	Start	Stop	Start	Stop	Start	Stop	
		4,000 RPM Aluminum Upmilling	1/4	h1	4.03E+05	4.16E+05	4.12E+05	4.25E+05		
h2	4.17E+05			4.31E+05	4.28E+05	4.36E+05				
h3	4.32E+05			4.40E+05	4.42E+05	4.50E+05				
h4	4.41E+05			4.48E+05	4.52E+05	4.57E+05				
1/2	h1		4.09E+05	4.26E+05	4.14E+05	4.30E+05	7.11E+05	7.41E+05	7.16E+05	7.46E+05
	h2		4.28E+05	4.45E+05	4.34E+05	4.49E+05	7.46E+05	7.76E+05	7.51E+05	7.80E+05
	h3		4.47E+05	4.58E+05	4.55E+05	4.63E+05	7.82E+05	8.00E+05	7.84E+05	8.02E+05
	h4		4.60E+05	4.70E+05	4.65E+05	4.74E+05	8.04E+05	8.18E+05	8.07E+05	8.23E+05
3/4	h1		5.15E+05	5.37E+05	5.46E+05	5.64E+05	9.09E+05	9.45E+05	9.12E+05	9.47E+05
	h2		5.40E+05	5.58E+05	5.62E+05	5.86E+05	9.51E+05	9.83E+05	9.50E+05	9.86E+05
	h3		5.69E+05	5.73E+05	5.88E+05	6.00E+05	9.86E+05	1.01E+06	9.91E+05	1.01E+06
	h4		5.75E+05	5.85E+05	6.02E+05	6.12E+05	1.01E+06	1.03E+06	1.02E+06	1.03E+06

For each block of data, the “findpeaks.m” function was used to identify the data index corresponding to the peak of each force cycle.

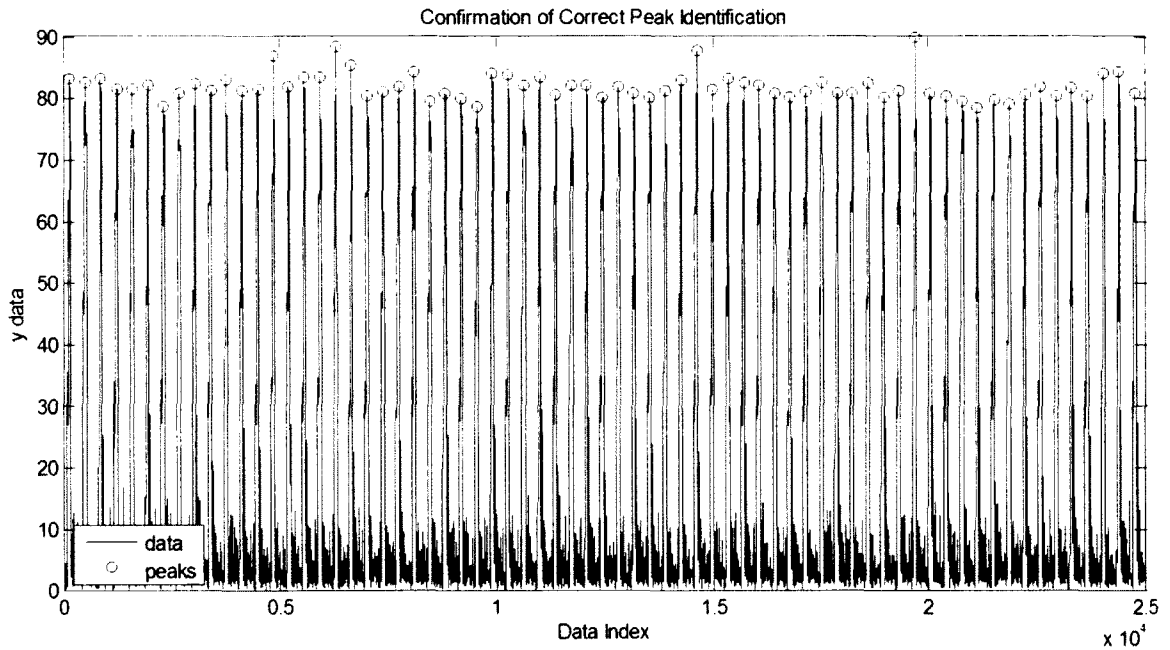


Figure D.4 - Identify peaks in the data set

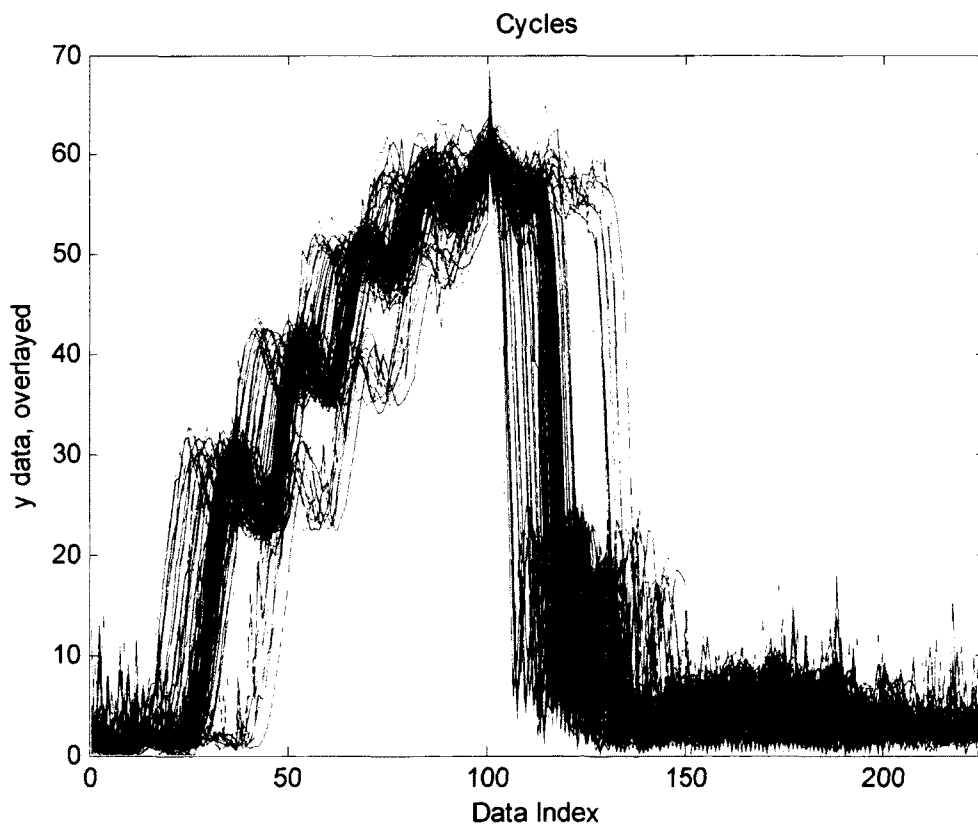


Figure D.5 - Cycles aligned by peak value

These cycles were then aligned by their peak value as shown in Figure D.5. To get an average waveform for each block of data, the correlation coefficient was used to determine the lag index for each cycle corresponding to best alignment.

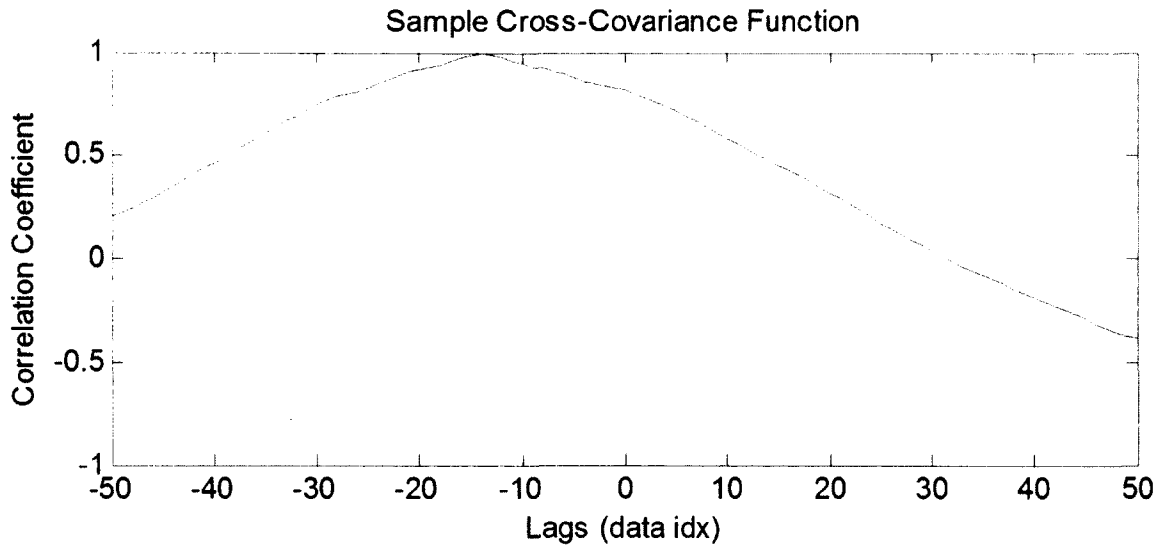


Figure D.6 - Cross-covariance definition of the correlation coefficient used to align data

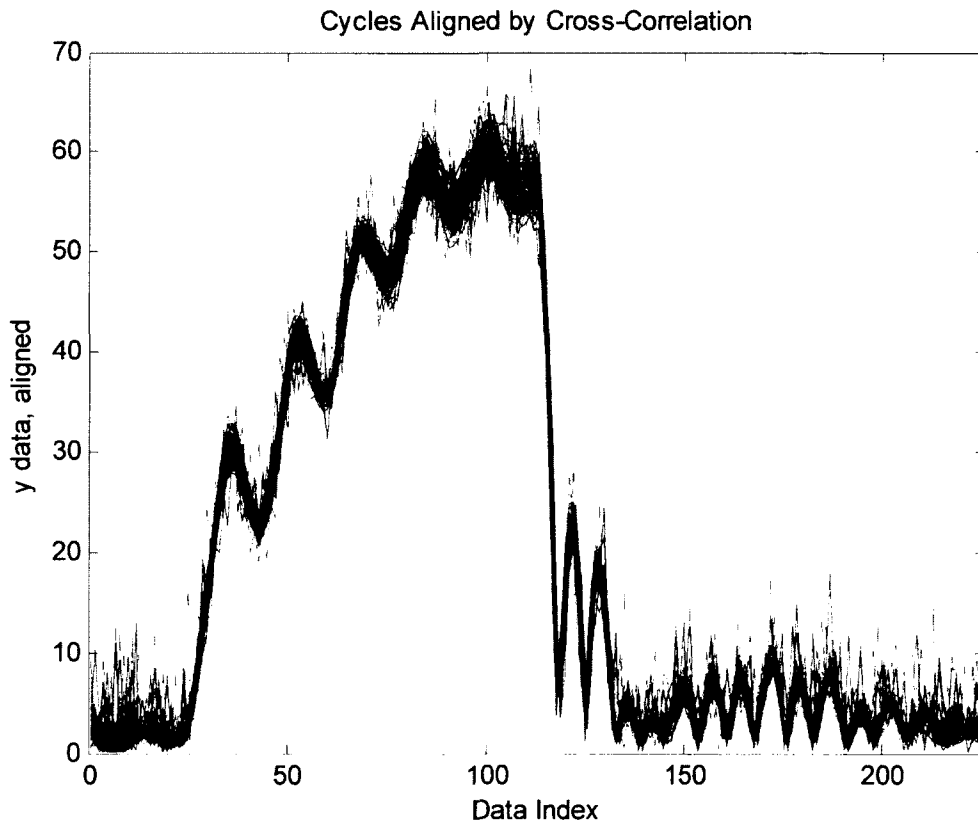


Figure D.7 - Aligned cycles are overlaid

Finally, the cycles were ensemble-averaged to obtain a characteristic profile for each cutting condition. The Kistler data was also averaged this way. This technique is known in DSP as coherent averaging.

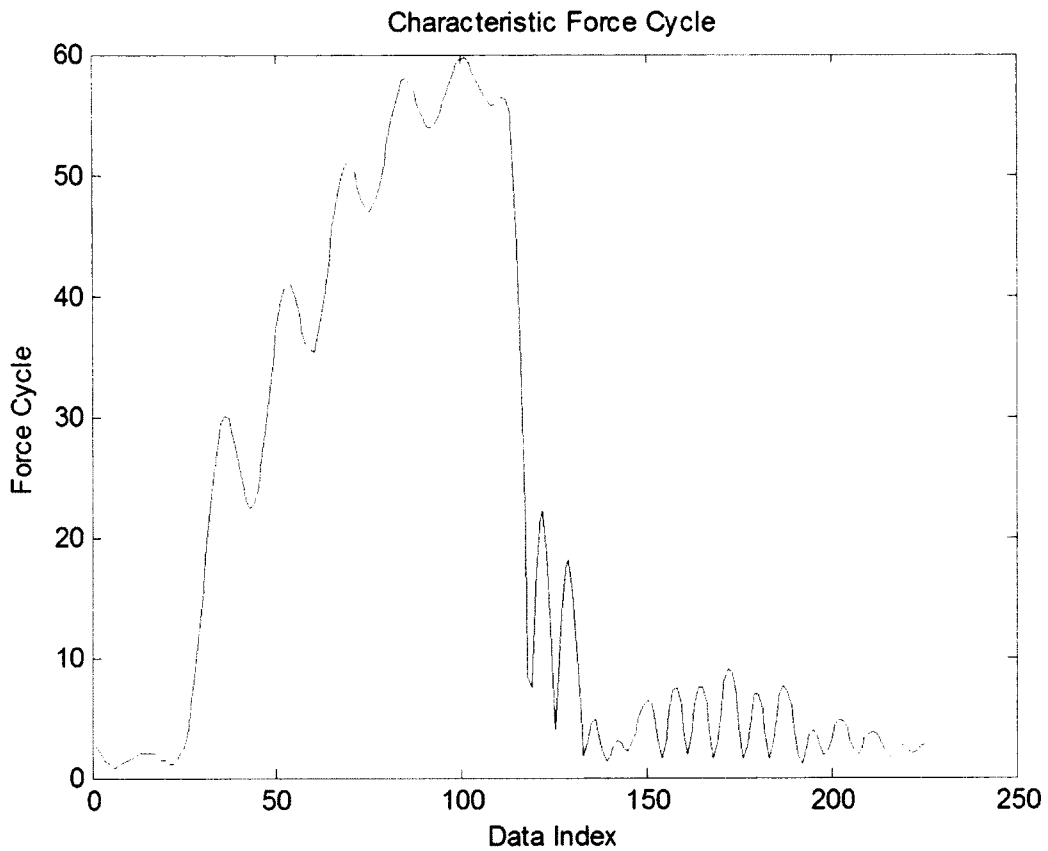


Figure D. 8 - Aligned cycles are ensemble averaged to obtain the net profile

The code used to process the Smart Tool data and Kistler data is presented below.

```

%% Net Force Comparison
% For Dec 15 Cutting tests
% Andrew Harmon

clear
close all
clc

load preprocessing.mat
load NetForce.mat

%% Kistler Data Pre-Processing
speed = {'rpm3000' 'rpm3600' 'rpm4000'};
namespace = {'kqtan' 'kqrad' 'khtan' 'khrad' 'k3qtan' 'k3qrad'};
chip = {'h1' 'h2' 'h3' 'h4'};

```

```

for i = 1:3 %RPM Select % Select 3000, 3600, 4000 rpm
    for j = 1:length(namespace) %Immersion Select % Select 1/4, 1/2, 3/4 immersion
        % Get Source
        source = eval( strcat('ksig.',namespace{j}) );
        for k = 1:4 %Feedrate Select % Select h1, h2, h3, h4 feedrate
            % Get Data
            switch rem(j,2)
                case 1 %tan
                    col = 5;
                case 0 %rad
                    col = 7;
            end
            idx1 = table((12*i-11)+((ceil(j/2)-1)*4)+(k-1),col); % Table indexing
            idx2 = table((12*i-11)+((ceil(j/2)-1)*4)+(k-1),col+1);
            if idx1~=0 && idx2~=0

                % Data Set of Interest
                data = source(idx1:idx2);

                % Get Average Waveform
                signals.(speed{i}).(namespace{j}).(chip{k}) = meanCycle(data);

                disp(strcat('signals.',speed{i},',' namespace{j},',' chip{k},': Done!'));
                pause;

            end
        end
    end
end

end

end

end

%% Smart Tool Data Pre-Processing
speed = {'rpm3000' 'rpm3600' 'rpm4000'};
namespace = {'sqtan' 'sgrad' 'shtan' 'shrad' 's3qtan' 's3qgrad'};
chip = {'h1' 'h2' 'h3' 'h4'};

% Process Tangential and Radial Data Independently
for i = 1:3 %RPM Select % Select 3000, 3600, 4000 rpm
    for j = 1:length(namespace) %Immersion Select % Select 1/4, 1/2, 3/4 immersion
        % Get Source
        source = eval( strcat('sig.',namespace{j}) );
        for k = 1:4 %Feedrate Select % Select h1, h2, h3, h4 feedrate
            % Get Data
            switch rem(j,2)
                case 1 %tan
                    col = 1;
                case 0 %rad
                    col = 3;
            end
            idx1 = table((12*i-11)+((ceil(j/2)-1)*4)+(k-1),col); % Table indexing
            idx2 = table((12*i-11)+((ceil(j/2)-1)*4)+(k-1),col+1);
            if idx1~=0 && idx2~=0

                % Data Set of Interest
                data = source(idx1:idx2);

                % Zero and scale the data
                force = bits2force(data,i,j);

                % Get Average Waveform
                signals.(speed{i}).(namespace{j}).(chip{k}) = meanCycle(force);

                disp(strcat('signals.',speed{i},',' namespace{j},',' chip{k},': Done!'));
                pause;

            end
        end
    end
end

end

end

namespace2 = {'sqnet' 'shnet' 's3qnet'};

```

```

Convert Radial and Tangential to Net Force
for i = 1:3 %RPM Select
    for j = [1 3 5] %Immersion Select
        for k = 1:4 %Feedrate Select

            disp(strcat('Starting:signals.',speed{i},'.', ...
                namespace2(ceil(j/2)),'.',chip{k}));

            tf1 = isfield(signals.(speed{i}),char(namespace2(ceil(j/2))));
            tf2 = 1;
            if tf1==1
                tf2 = isfield(signals.(speed{i}).(namespace2(ceil(j/2))), ...
                    char(chip{k}));
            end

            if tf1==0 || tf2==0

                % Data Set of Interest
                tan = signals.(speed{i}).(namespace{j}).(chip{k});
                rad = signals.(speed{i}).(namespace{j+1}).(chip{k});

                % Ignore Negative Data
                for count = 1:length(tan)
                    if tan(count) <= 0
                        tan(count) = 0;
                    end
                end
                for count = 1:length(rad)
                    if rad(count) <= 0
                        rad(count) = 0;
                    end
                end

                % Align the Data Sets
                figure(1)
                plot(tan,'b.-');
                hold on;
                plot(rad,'r.-');

                satisfied = 0;
                while satisfied == 0
                    maxlag = input('        Number of points to shift the data? >>');

                    tan = circshift(tan,[0,maxlag]);

                    % Plot the data
                    figure(1);
                    hold off
                    plot(tan,'b.-');
                    hold on
                    plot(rad,'r.-');
                    title('Cycles Aligned by Circshift')
                    xlabel('Data Index')
                    ylabel('Force Cycles, aligned')

                    satisfied = input('        Are you satisfied with the alignment?
(Y=1, N=0) >>');
                end

                % Compute Net Force
                last = min(length(tan),length(rad));
                netForce = sqrt( tan(1:last).^2 + rad (1:last).^2 );
                for count = 1:length(tan)
                    if tan(count) <= 0
                        tan(count) = 0;
                    end
                end

                figure(1);
                hold on
                plot(netForce,'g.-');
            end
        end
    end
end

```



```

        title('Characteristic Force Cycle')
        xlabel('Data Index')
        ylabel('Force Cycle')

        satisfied = 0;
        while satisfied == 0
            shiftidx = input('      Number of lags to shift the averaged
            waveform >>');
            netForce = circshift(netForce,[0 -shiftidx]);

            figure(1);
            hold off
            plot(netForce,'g.-');
            title('Characteristic Force Cycle')
            xlabel('Data Index')
            ylabel('Force Cycle')

            satisfied = input('Are you satisfied with the alignment? (Y=1,
            N=0) >>');
        end

        % Crop Dataset
        stop = input('      Crop Stop Index >>');
        netForce = netForce(1:stop);
        signals.(speed{i}).(namespace2{ceil(j/2)}).(chip{k}) = netForce;
        figure(1);
        hold off
        plot(netForce,'g.-');
        title('Characteristic Force Cycle')
        xlabel('Data Index')
        ylabel('Force Cycle')

        disp(strcat('
        signals.',speed{i},',' namespace2{ceil(j/2)},',' chip{k},': Done!'));
        pause(1)
    else
        disp('      Redundant Calculation: Data Skipped!')
    end
end
end
end

end
end

%% Plot all on Same Graph

kfs = 18018; % Kistler sampling frequency
fs = 10240; % Smart Tool sampling frequency

speed = {'rpm3000' 'rpm3600' 'rpm4000'};
immersion = {'1/4' '1/2' '3/4'};
namespace1 = {'kqtan' 'khtan' 'k3qtan'};
namespace2 = {'kqrad' 'khrad' 'k3qrad'};
namespace3 = {'sqnet' 'shnet' 's3qnet'};
chip = {'h1' 'h2' 'h3' 'h4'};
frame = [1 1 1 1 2 2 2 2 3 3 3 3 4 4 4 4 5 5 5 5 6 6 6 6 7 7 7 7 8 8 8 8 9 9 9 9];
sequence = [1 2 3 4 1 2 3 4 1 2 3 4 1 2 3 4 1 2 3 4 1 2 3 4 1 2 3 4 1 2 3 4 1 2 3 4];

count = 0;
for i = 1:3
    for j = 1:3
        for k = 1:4

            % Get Kistler Experiment 1 Profile
            tf1 = isfield(signals.(speed{i}),char(namespace1{j}));
            tf2 = 0;
            if tf1==1
                tf2 = isfield(signals.(speed{i}).(namespace1{j}),char(chip{k}));
            end
            if tf1==1 && tf2==1
                K1 = signals.(speed{i}).(namespace1{j}).(chip{k});
            else
                K1 = [0 0];
            end
        end
    end
end

```

```

end

% Get Kistler Experiment 2 Profile
tf1 = isfield(signals.(speed{i}),char(namespace2{j}));
tf2 = 0;
if tf1==1
    tf2 = isfield(signals.(speed{i}).(namespace2{j}),char(chip{k}));
end
if tf1==1 && tf2==1
    K2 = signals.(speed{i}).(namespace2{j}).(chip{k});
else
    K2 = [0 0];
end

% Get Smart Tool Profile
ST = signals.(speed{i}).(namespace3{j}).(chip{k});

%%

% Time Vectors and Errata
ktime1 = (0:length(K1)-1)./kfs;
ktime2 = (0:length(K2)-1)./kfs;

% Interpolate Onto Common Time Axis
profile.K1 = interp1(ktime1,K1,0:1/fs:max(ktime1));
profile.K2 = interp1(ktime2,K2,0:1/fs:max(ktime2));

% Zero Pad Smaller Vectors
N = max([length(profile.K1) length(profile.K2) length(profile.ST)]);
time = (0:N-1)./fs;
signalspace = {'K1' 'K2' 'ST'};
for a=1:3
    n = length(profile.(signalspace{a}));
    if n~N
        profile.(signalspace{a}) = [profile.(signalspace{a}), zeros(1,N-n)];
    end
end

%%

% Plot Data
hold off
count = count+1;
figure(frame(count));
subplot(2,2,sequence(count))
hold off
subplot(2,2,k)
N = length(K1);
time = (0:N-1)./fs*1000;
plot(time,K1,'bo-')
hold on
plot(time,K2,'r.-')
plot(time,ST,'g*-')
title(strcat('signals.',speed{i},','.',immersion{j},','.',chip{k}));
xlabel('Time (ms)')
ylabel('Net Force (N)')
if k==1
    legend('Kistler 1','Kistler 2','Smart Tool','location','northwest')
end

xlim([0 10])
ylim([0 500])
legend('Kistler 1','Kistler 2','Smart Tool','location','northwest')
grid on

%%

% Plot Misaligned Data
hold off
figure(1);
plot(profile.K1,'bo-')
hold on
plot(profile.K2,'r.-')
plot(profile.ST,'g*-')
title(strcat('signals.',speed{i},','.',immersion{j},','.',chip{k}));
xlabel('Time (s)')

```

```

        ylabel('Net Force (N)')
        legend('Kistler 1','Kistler 2','Smart Tool','location','northwest')

% Circshift K1 on Plot
satisfied = 0;
while satisfied == 0
    shiftidx = input('Number of lags to shift K1 >>');
    profile.K1 = circshift(profile.K1,[0 shiftidx]);

    % Plot Misaligned Data
    hold off
    figure(1);
    plot(profile.K1,'bo-')
    hold on
    plot(profile.K2,'r.-')
    plot(profile.ST,'g*-')
    title(strcat('signals.',speed{i},'.',immersion{j},'.',chip{k}));
    xlabel('Time (s)')
    ylabel('Net Force (N)')
    legend('Kistler 1','Kistler 2','Smart Tool','location','northwest')

    satisfied = input('Are you satisfied with the alignment? (Y=1, N=0) >>');
end

% Circshift K2 on Plot
satisfied = 0;
while satisfied == 0
    shiftidx = input('Number of lags to shift K2 >>');
    profile.K2 = circshift(profile.K2,[0 shiftidx]);

    % Plot Misaligned Data
    hold off
    figure(1);
    plot(profile.K1,'bo-')
    hold on
    plot(profile.K2,'r.-')
    plot(profile.ST,'g*-')
    title(strcat('signals.',speed{i},'.',immersion{j},'.',chip{k}));
    xlabel('Time (s)')
    ylabel('Net Force (N)')
    legend('Kistler 1','Kistler 2','Smart Tool','location','northwest')

    satisfied = input('Are you satisfied with the alignment? (Y=1, N=0) >>');
end

% Circshift ST on Plot
satisfied = 0;
while satisfied == 0
    shiftidx = input('Number of lags to shift ST >>');
    profile.ST = circshift(profile.ST,[0 shiftidx]);

    % Plot Misaligned Data
    hold off
    figure(1);
    plot(profile.K1,'bo-')
    hold on
    plot(profile.K2,'r.-')
    plot(profile.ST,'g*-')
    title(strcat('signals.',speed{i},'.',immersion{j},'.',chip{k}));
    xlabel('Time (s)')
    ylabel('Net Force (N)')
    legend('Kistler 1','Kistler 2','Smart Tool','location','northwest')

    satisfied = input('Are you satisfied with the alignment? (Y=1, N=0) >>');
end

% Trim Data Record
stop = input('Crop Stop Index >>');
profile.K1 = profile.K1(1:stop);
profile.K2 = profile.K2(1:stop);
profile.ST = profile.ST(1:stop);

```

```

        % Save Aligned Results
        signals.(speed{i}).(namespace1{j}).(chip{k}) = profile.K1;
        signals.(speed{i}).(namespace2{j}).(chip{k}) = profile.K2;
        signals.(speed{i}).(namespace3{j}).(chip{k}) = profile.ST;
    end
end
end

```

```

function smoothed = meanCycle(ydata)

%% Locate Peaks

% % Prompt for minimum height and distance
% minpeakheight = input('Minimum Peak Height? >>');
% minpeakdistance = input('Minimum Peak Distance? >>');

minpeakheight = 1.4*mean(ydata);
minpeakdistance = 100;

% Locate the peaks
[pks, locs] = findpeaks(ydata,'minpeakheight',minpeakheight,...
    'minpeakdistance',minpeakdistance);

pks = pks(2:length(pks)-1);
locs = locs(2:length(locs)-1);

% Confirm correct peak identification
figure(1)
plot(ydata,'b-');
hold on
plot(locs,pks,'ko');
title('Confirmation of Correct Peak Identification')
xlabel('Data Index')
ylabel('y data')
xlim([0 length(ydata)]);
legend('data','peaks','location','southwest');

%% Capture Traces

before = 60;
after = 50;

satisfied = 0;
while satisfied == 0

% Sort cycles into row vectors
    out = zeros(length(pks)-1,before+after+1);
    for n = 1:length(pks)-1;
        out(n,:) = ydata((locs(n)-before):(locs(n)+after));
    end

% % Normalize traces
%     for i=1:length(pks)
%         out(i,:) = (out(i,:)-mean(out(i,:)))/max(out(i,:)-mean(out(i,:)));
%     end

% Plot the data
    figure(1);
    hold off
    plot(out');
    title('Cycles')
    xlabel('Data Index')
    ylabel('y data, overlaid')
    xlim([0 before+after+1]);

```

```

    satisfied = input('Are you satisfied with the capture? (Y=1, N=0) >>');
    if satisfied == 0
        % Prompt for minimum height and distance
        before = input('Number of points to keep before peak? >>');
        after = input('Number of points to keep after peak? >>');
    end
end

%% Align the Traces

satisfied = 0;
while satisfied == 0
    maxlag = input('Number of points to shift the cross-correlation? >>');

    % Use a cross-correlation to align the data
    out2 = zeros(size(out));
    for i=1:length(pks)-1
        cor = ccorr(out(1,:),out(i,:),-maxlag:maxlag);
        [~, idx] = max(cor.C);
        out2(i,:) = circshift(out(i,:),[0,idx+(maxlag+1)]);
    end

    % Matlab's Cross-Correlation
    % out2 = zeros(size(out));
    % for i=2:length(pks)-2
    %     [C lags] = xcorr(out(:,1),out(:,i),maxlag);
    %     [~, idx] = max(C);
    %     out2(i,:) = circshift(out(i,:),[0,(idx+(maxlag+1))]);
    %     disp(idx)
    % end

    % Plot the data
    figure(1);
    hold off
    plot(out2');
    title('Cycles Aligned by Cross-Correlation')
    xlabel('Data Index')
    ylabel('y data, aligned')
    xlim([0 before+after+1])

    satisfied = input('Are you satisfied with the correlation? (Y=1, N=0) >>');
end

%% Average the Cycles

smoothed = mean(out2);

figure(1);
hold off
plot(smoothed,'b-');
title('Characteristic Force Cycle')
xlabel('Data Index')
ylabel('Force Cycle')

satisfied = 0;
while satisfied == 0
    shiftidx = input('Number of lags to shift the averaged waveform >>');
    smoothed = circshift(smoothed,[0 shiftidx]);

    figure(1);
    hold off
    plot(smoothed,'b-');
    title('Characteristic Force Cycle')
    xlabel('Data Index')
    ylabel('Force Cycle')

    satisfied = input('Are you satisfied with the alignment? (Y=1, N=0) >>');
end

```

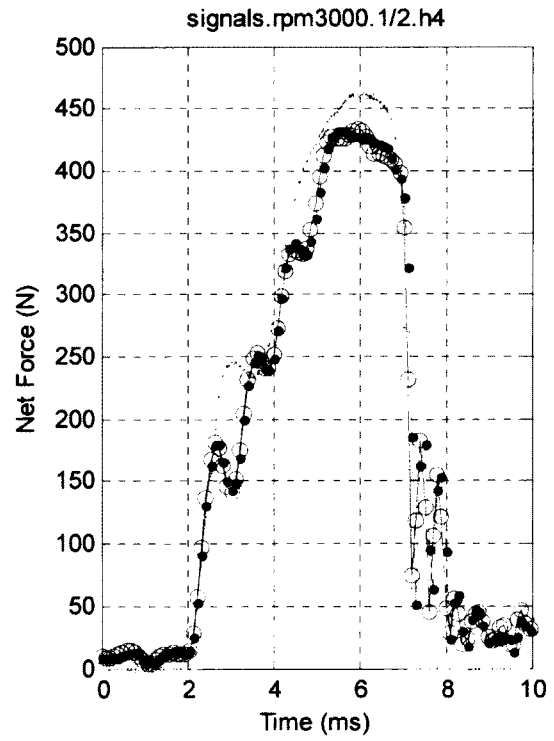
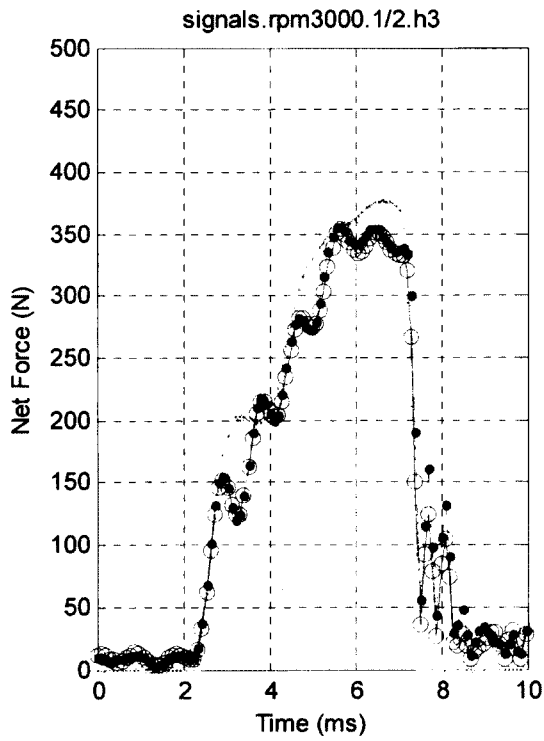
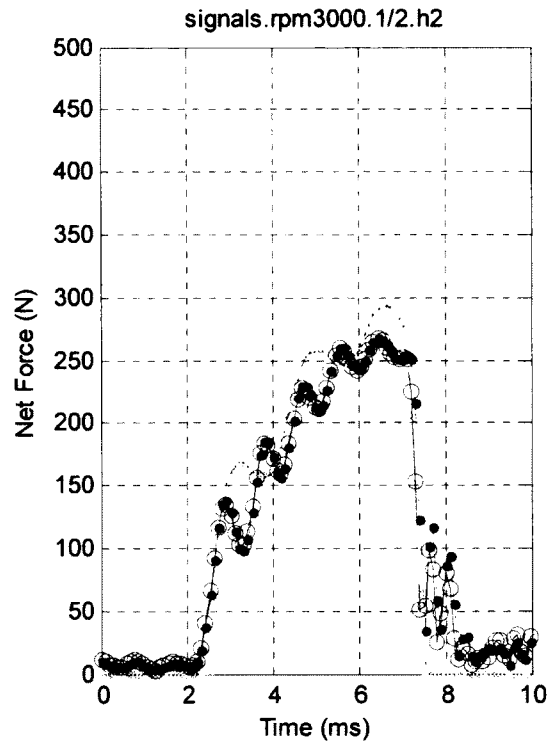
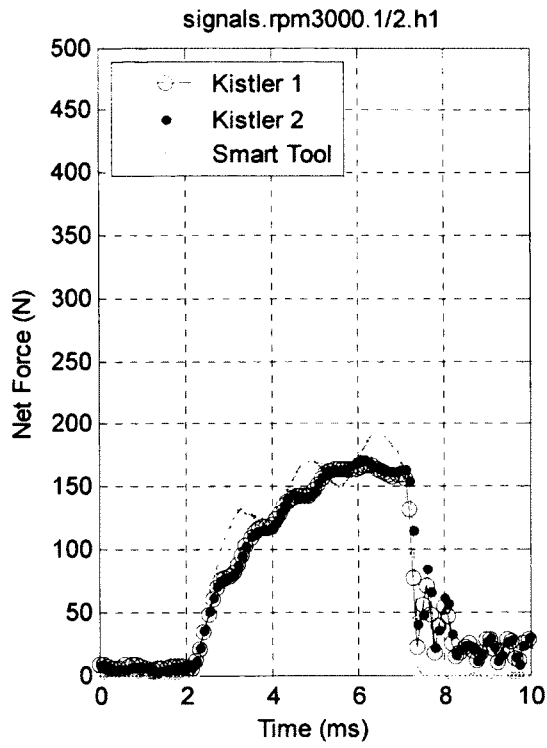


Figure D.9 - Net force profile comparison: 3000 rpm, half immersion

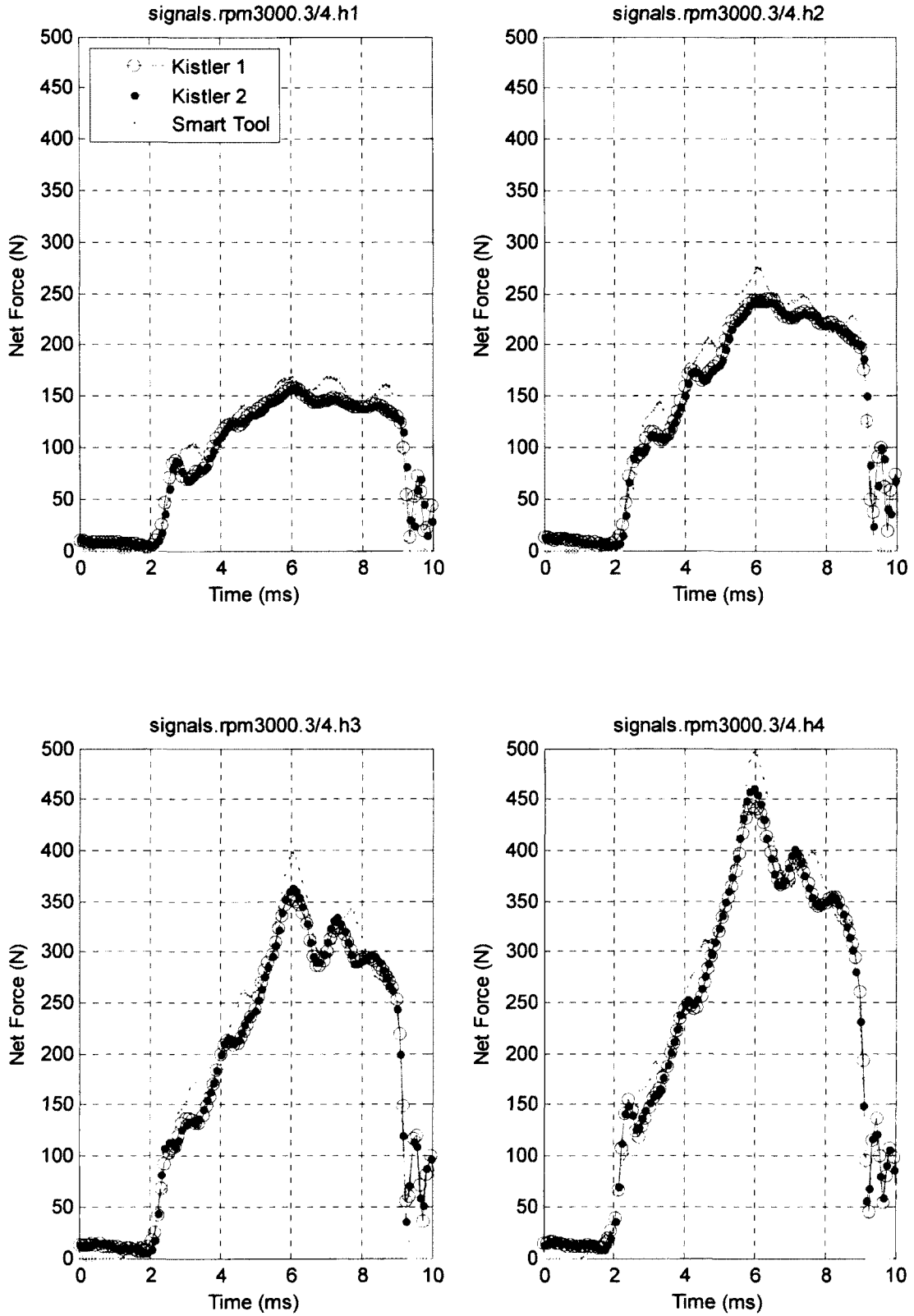


Figure D.10 – Net force profile comparison: 3000 rpm, three-quarter immersion

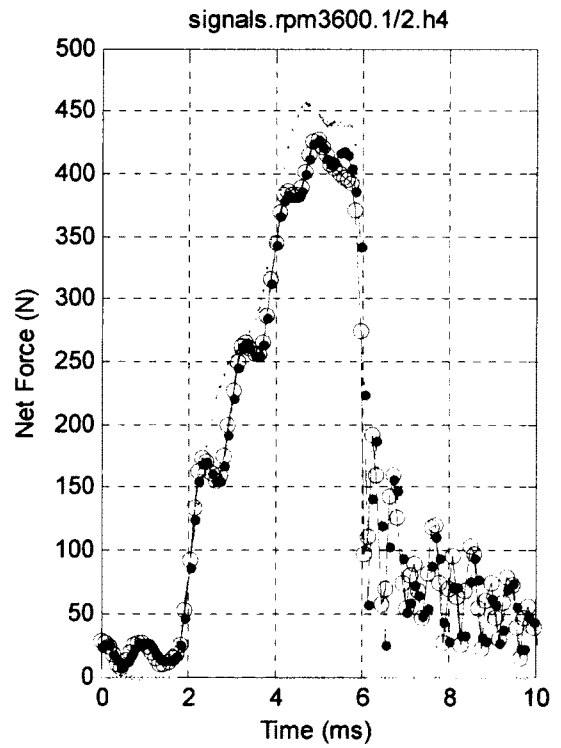
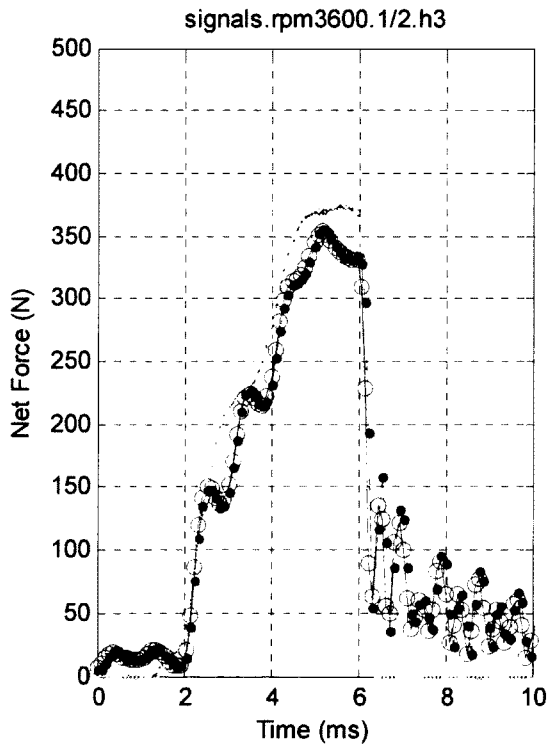
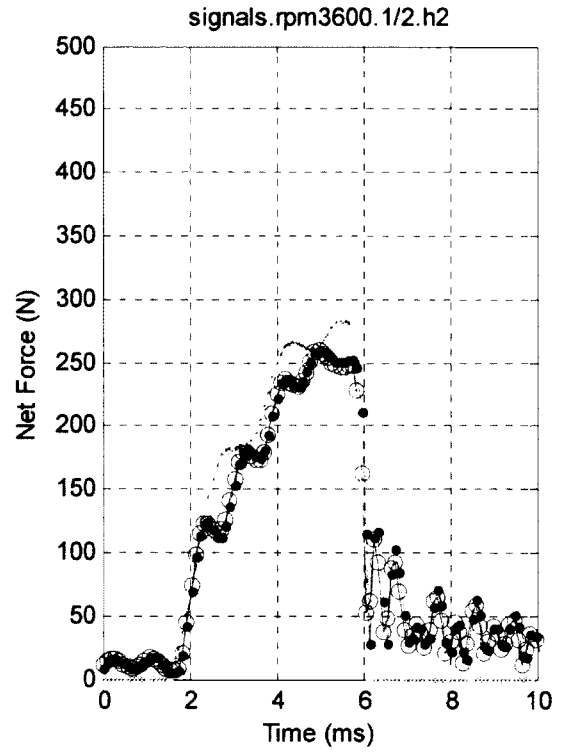
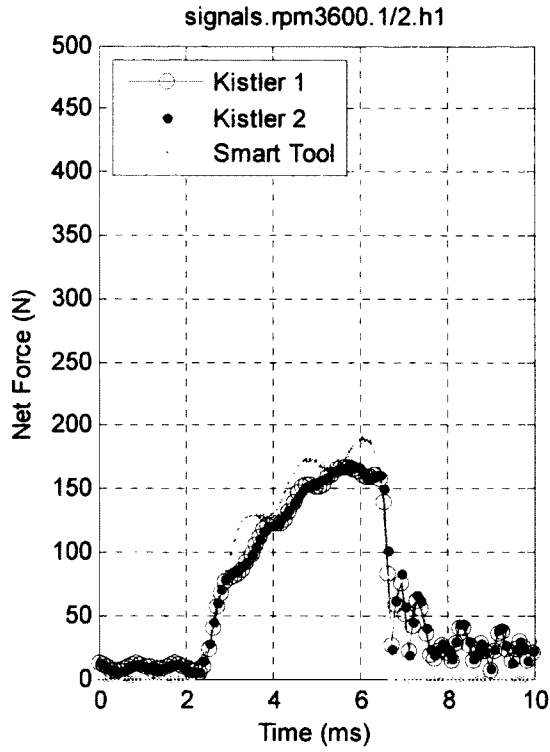


Figure D.11 - Net force profile comparison: 3600 rpm, half immersion

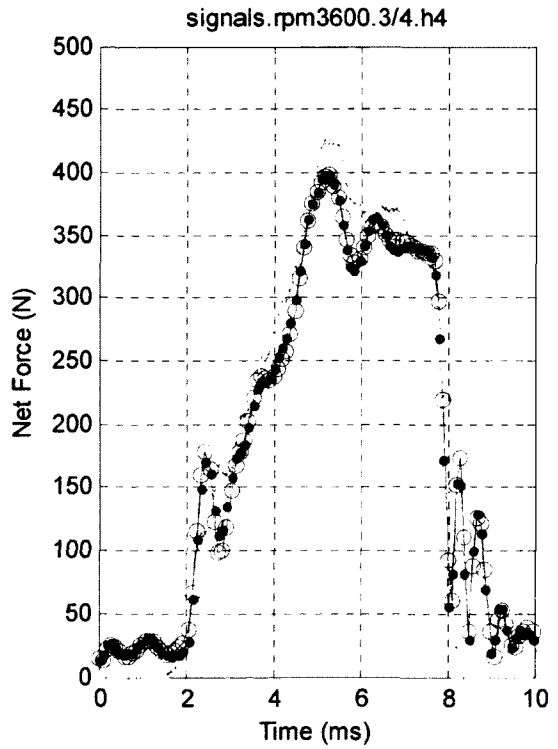
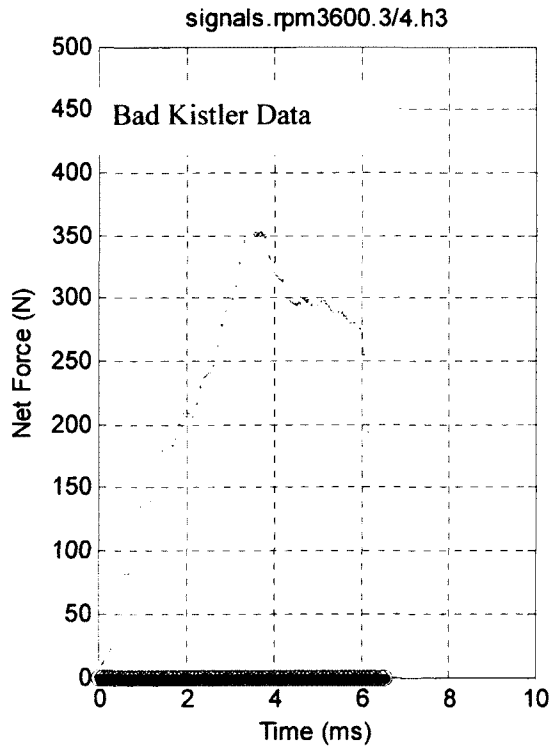
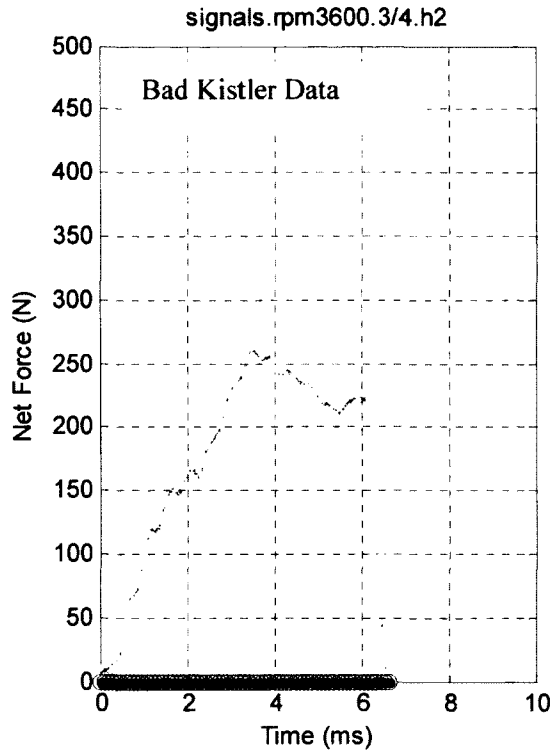
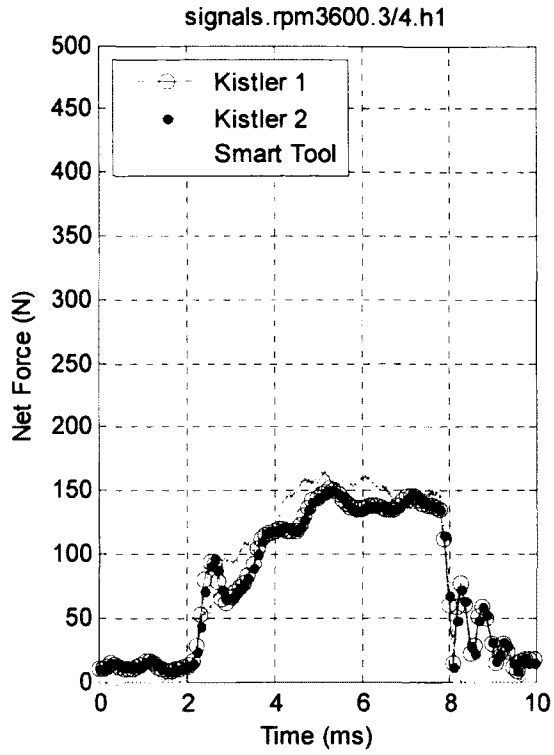


Figure D.12 - Net force profile comparison: 3600 rpm, three-quarter immersion

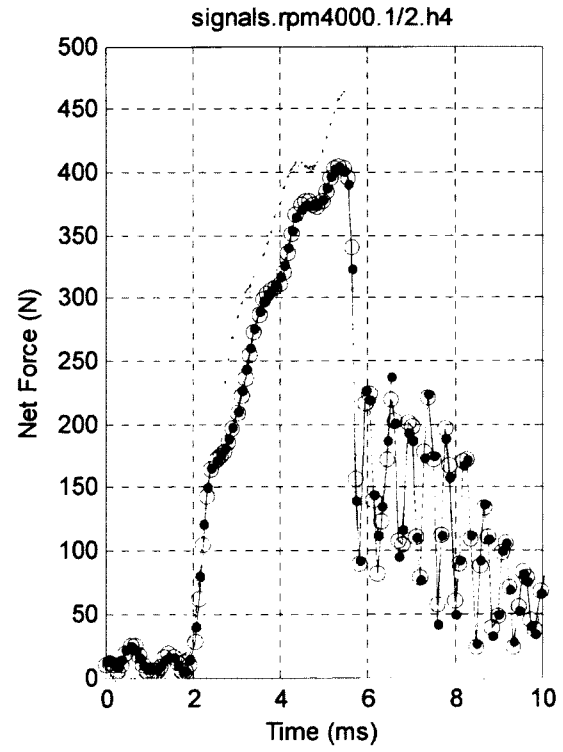
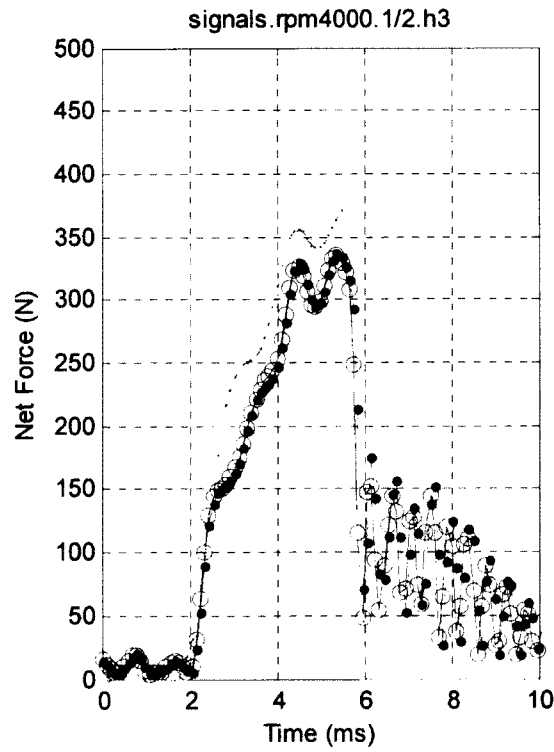
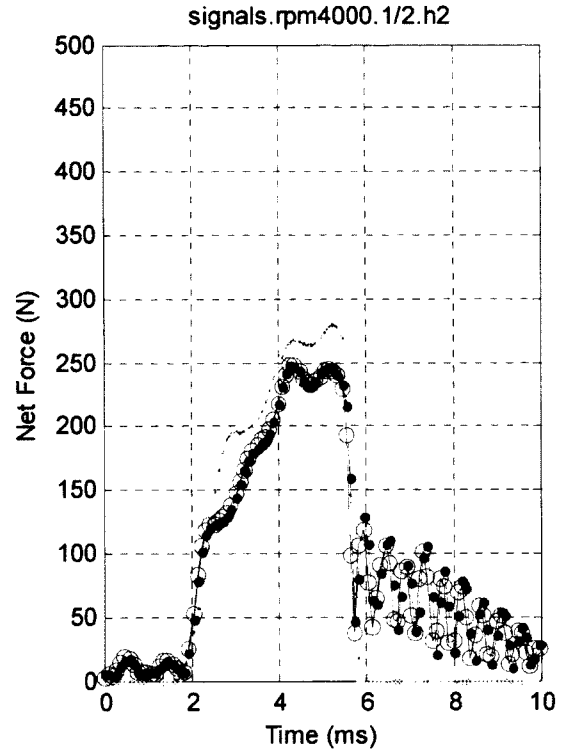
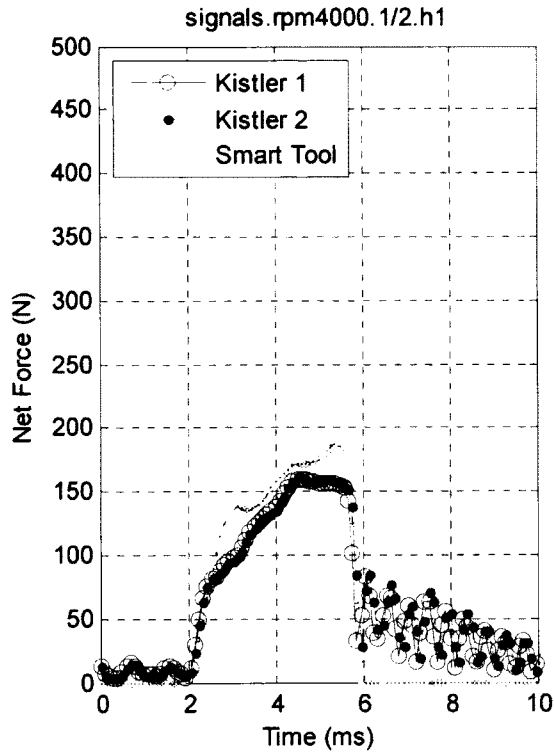


Figure D.13 - Net force profile comparison: 4000 rpm, half immersion

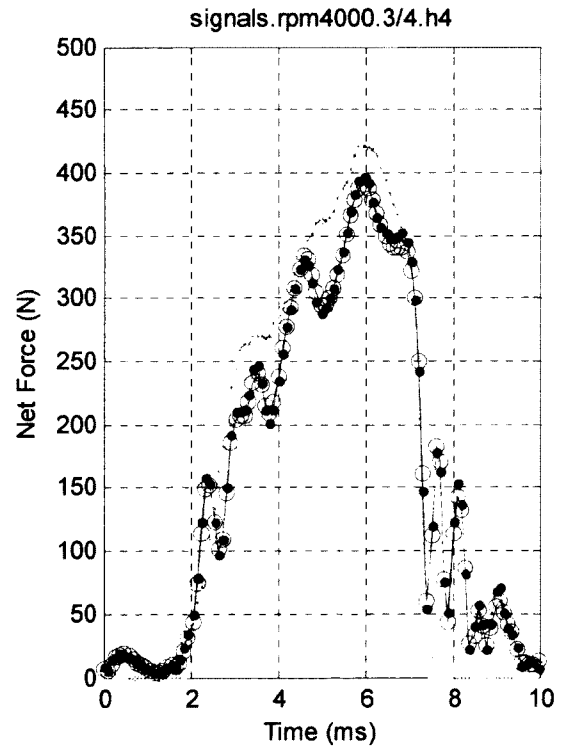
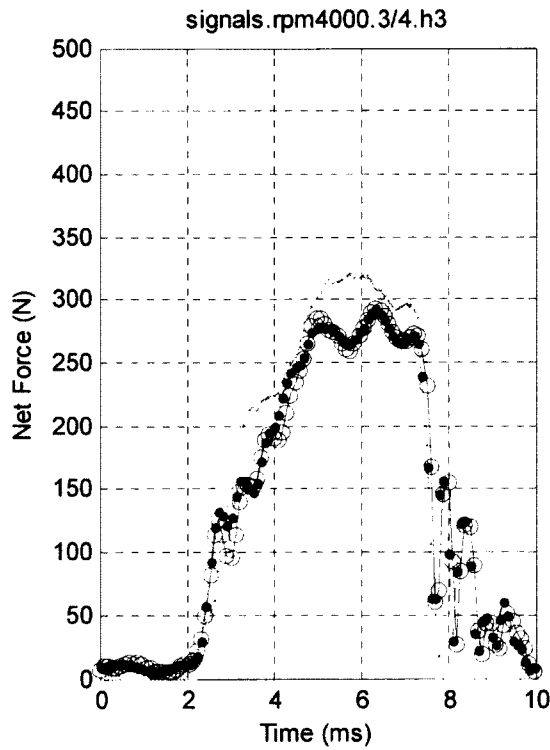
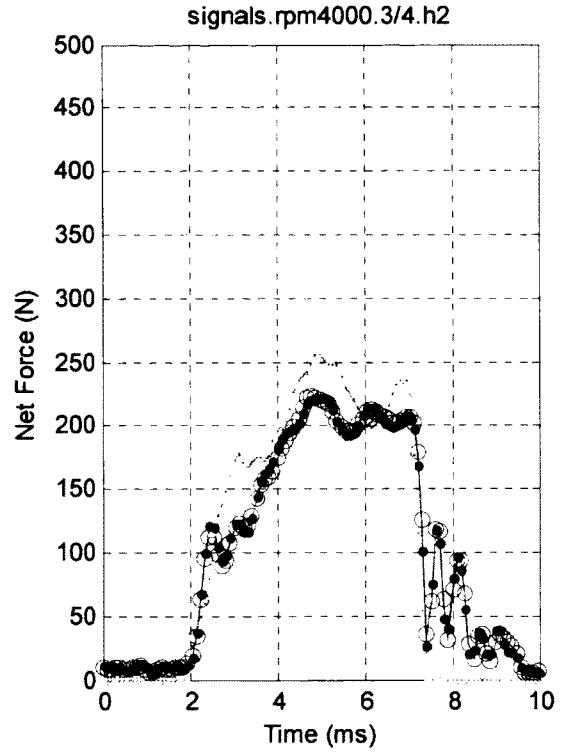
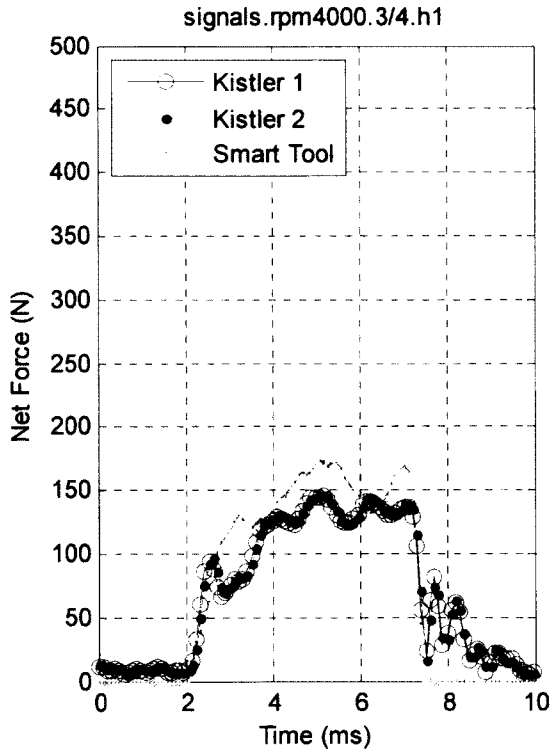


Figure D.14 - Net force profile comparison: 4000 rpm, three-quarter immersion

APPENDIX E

ARCHIVE OF MATLAB M-FILES

File for Facilitating Data Collection with Smart Tool v.10

```
clear;
clc;
close all;

Record_Length = 70; % seconds
try

% clean up existing serial components in case the previous run died
instrs = instrfind;
if (~isempty(instrs))
    disp('Cleaning old serial ports...');
    %fclose(instrs);
    delete(instrs);
end

disp('Allocating serial port...');

% set up the com port
s_port = serial('COM40');
set(s_port, 'BaudRate', 8*115200);
set(s_port, 'Timeout', 3);
bufferSize = 2^12;
set(s_port, 'InputBufferSize', bufferSize);

disp('Trying to open serial port...');

% open the port
fopen(s_port);

disp('Opened!');
disp('Configuring Toolholder...');
fwrite(s_port, '$$$');
pause(1);
fwrite(s_port, sprintf('F,1\n'));
fwrite(s_port, 'E');
fwrite(s_port, '0');
pause(0.2);
fwrite(s_port, '9');

pause(1);
if(s_port.BytesAvailable>0)
control_signal = fread(s_port, s_port.BytesAvailable);
```

```

end
fwrite(s_port, 'G');
disp('Finished Configuration');

strain_sample_rate = 10240;
temp_interval = 4*2*3000;
next_temp = 0;
temp_sample_rate = strain_sample_rate/(temp_interval/2);

strain_display = strain_sample_rate*Record_Length;
temp_display = ceil(temp_sample_rate*60);

strain = NaN.*ones(1, strain_display*2);
strain_index = 1;
temp = zeros(1, temp_display*2);
temp_index = 1;
raw = zeros(1, 100000);
raw_index = 1;

strain_time = linspace(0, (strain_display-
1)/strain_sample_rate, strain_display);
temp_time = linspace((1-temp_display)/temp_sample_rate, 0,
temp_display);

disp('Collecting data...');
systemsound('Windows XP Print Complete');
tic
while sum(isnan(strain)) > 0
    bytes = fread(s_port, max(100, floor(s_port.BytesAvailable/2)*2)); %
read in multiples of 2
    if (length(bytes) == bufferSize)
        warning('May have lost data. Attempting to resynchronize.');
```

```

        fwrite(s_port, 'Q');
        pause(1);
        if (s_port.BytesAvailable);
            fread(s_port, s_port.BytesAvailable);
        end
        next_temp = 0;
        fwrite(s_port, 'G');
    elseif (isempty(bytes))
        error('No data!');
    else
        raw(raw_index:raw_index+length(bytes)-1) = bytes;
        raw_index = raw_index + length(bytes);
        if (raw_index > 75000)
            raw_index = 1;
        end

        % extract temperature if it's present
        if (length(bytes) > next_temp)
            temp_index:temp_index+1) = ...
                bytes(next_temp+1:next_temp+2);
            bytes(next_temp+1:next_temp+2) = [];
            temp_index = temp_index + 2;
            if (temp_index > temp_display*2)
                temp_index = 1;
            end
        end
    end
end

```

```

        next_temp = temp_interval + next_temp - length(bytes);

        fwrite(s_port, 'K');

    else
        next_temp = next_temp - length(bytes);
    end

    len = min(length(bytes), strain_display*2-strain_index+1);
    strain(strain_index:strain_index+len-1) = bytes(1:len);
    if (len == strain_display*2-strain_index+1)
        strain_index = length(bytes) - len + 1;
        strain(1:strain_index-1) = bytes(len+1:end);
    else
        strain_index = strain_index + len;
    end
end
end
disp('Data collection complete');
toc;
plot(strain_time, [s(i:end),s(1:i-1)]);
catch
    disp('Closing ports...');
    fclose(s_port);
    delete(s_port);
    disp('Closed!');
    rethrow(lasterror);
    close all;
end
end

```

Example Code for Linear Prediction Order Comparison

```
% Get Data (3/4 Immersion upmilling, 3000 rpm, rad, h2)
load Aluminum3k36k4k3QuarterUpRadial.mat

startIdx = 8.58e4;           % Starting index of data sequence
spr = 10240/3000*60;        % Samples per revolution
Nrev = 150;                 % Number of revolutions
stopIdx = round(startIdx + Nrev*spr);
force = (s(startIdx:stopIdx)-505)./1.64*4.448; % N
decay = force(343:430);    % Damped vibration profile
N = length(force);         % Length of data set
fs = 10240;                 % Sampling Frequency (Hz)
time = (0:N-1)./fs;        % Time vector (sec)

% Plot the force data
figure(1)
hold off
plot(time*1000,force);
title('3/4 Immersion Upmilling at 3000 RPM')
xlabel('time(ms)')
ylabel('Measured Force (N)')
xlim([0 100])

% Power Spectrum of the Whole Signal
[Pxx,W] = PWELCH(force,4096,3000,4096);
F1 = W/2/pi*fs;

figure(2)
hold off
semilogy(F1,Pxx/fs,'color',[0.5 0.5 0.5])
title('Welch Power Spectrum Estimate')
xlabel('Frequency (Hz)')
ylabel('Power Spectral Density (N^2/ Hz)')
xlim([0 5000])
ylim([5e-6 1e2])

% Linear Prediction Models
[a6 g6] = lpc(force,6);
[a12 g12] = lpc(force,12);
[a32 g32] = lpc(force,32);
[a64 g64] = lpc(force,64);
[a128 g128] = lpc(force,128);
[a256 g256] = lpc(force,256);

[H6 F] = freqz(1,a6,4096,'half',10240);
Pxx6 = g6*abs(H6).^2./2./fs;
figure(3)
hold off
semilogy(F1,Pxx./fs,'color',[0.65 0.65 0.65])
hold on
semilogy(F,Pxx6,'color',[0.7 0 0.7])
title('LPC Power Spectral Estimate, P=6')
xlabel('Frequency (Hz)')
ylabel('Power Spectral Density (N^2 /Hz)')
xlim([0 5000])
```

```

        legend('Welch PSD Estimate','LPC Estimate,
P=6','location','northeast')

[H12 F] = freqz(1,a12,4096,'half',10240);
Pxx12 = g12*abs(H12).^2./2./fs;
figure(4)
    hold off
    semilogy(F1,Pxx./fs,'color',[0.65 0.65 0.65])
    hold on
    semilogy(F,Pxx12,'color',[0.7 0 0.7])
    title('LPC Power Spectral Estimate, P=12')
    xlabel('Frequency (Hz)')
    ylabel('Power Spectral Density (N^2 /Hz)')
    xlim([0 5000])
    legend('Welch PSD Estimate','LPC Estimate,
P=12','location','northeast')

[H32 F] = freqz(1,a32,4096,'half',10240);
Pxx32 = g32*abs(H32).^2./2./fs;
figure(5)
    hold off
    semilogy(F1,Pxx./fs,'color',[0.65 0.65 0.65])
    hold on
    semilogy(F,Pxx32,'color',[0.7 0 0.7])
    title('LPC Power Spectral Estimate, P=32')
    xlabel('Frequency (Hz)')
    ylabel('Power Spectral Density (N^2 /Hz)')
    xlim([0 5000])
    legend('Welch PSD Estimate','LPC Estimate,
P=32','location','northeast')

[H64 F] = freqz(1,a64,4096,'half',10240);
Pxx64 = g64*abs(H64).^2./2./fs;
figure(6)
    hold off
    semilogy(F1,Pxx./fs,'color',[0.65 0.65 0.65])
    hold on
    semilogy(F,Pxx64,'color',[0.7 0 0.7])
    title('LPC Power Spectral Estimate, P=64')
    xlabel('Frequency (Hz)')
    ylabel('Power Spectral Density (N^2 /Hz)')
    xlim([0 5000])
    legend('Welch PSD Estimate','LPC Estimate,
P=64','location','northeast')

[H128 F] = freqz(1,a128,4096,'half',10240);
Pxx128 = g128*abs(H128).^2./2./fs;
figure(7)
    hold off
    semilogy(F1,Pxx./fs,'color',[0.65 0.65 0.65])
    hold on
    semilogy(F,Pxx128,'color',[0.7 0 0.7])
    title('LPC Power Spectral Estimate, P=128')
    xlabel('Frequency (Hz)')
    ylabel('Power Spectral Density (N^2 /Hz)')
    xlim([0 5000])

```



```

legend('Welch PSD Estimate','LPC Estimate,
P=128','location','northeast')

[H256 F] = freqz(1,a256,4096,'half',10240);
Pxx256 = g256*abs(H256).^2./2./fs;
figure(8)
hold off
semilogy(F1,Pxx./fs,'color',[0.65 0.65 0.65])
hold on
semilogy(F,Pxx256,'color',[0.7 0 0.7])
title('LPC Power Spectral Estimate, P=256')
xlabel('Frequency (Hz)')
ylabel('Power Spectral Density (N^2 /Hz)')
xlim([0 5000])
legend('Welch PSD Estimate','LPC Estimate,
P=256','location','northeast')

```

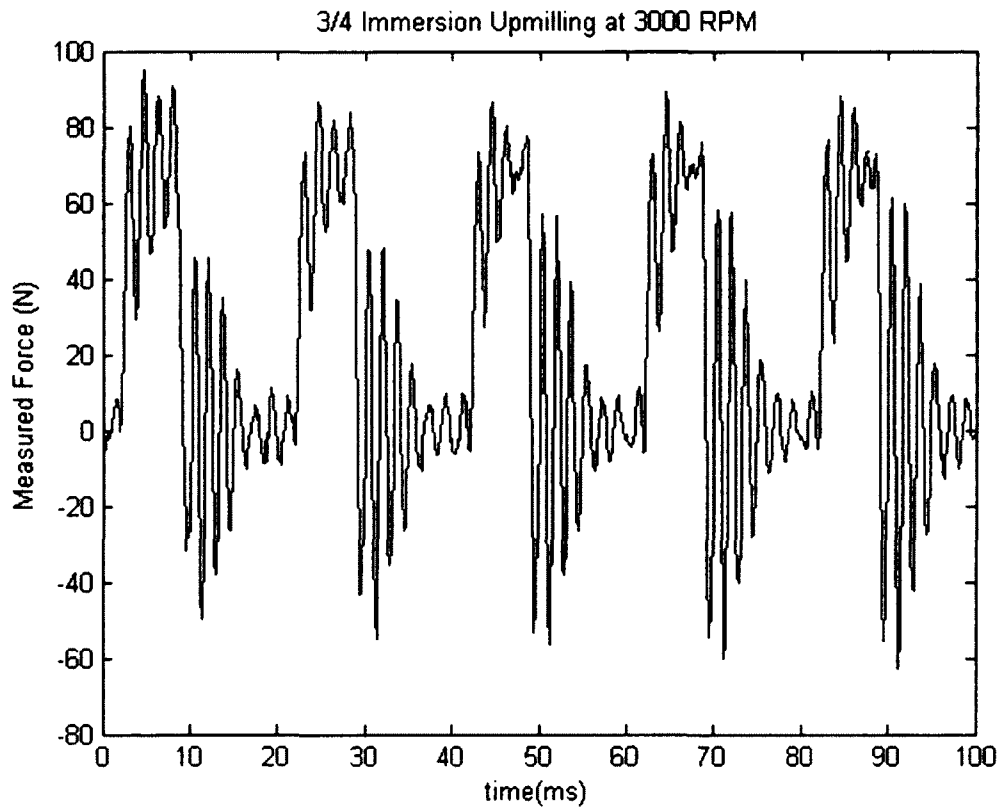


Figure E.1 – Radial strain converted to force

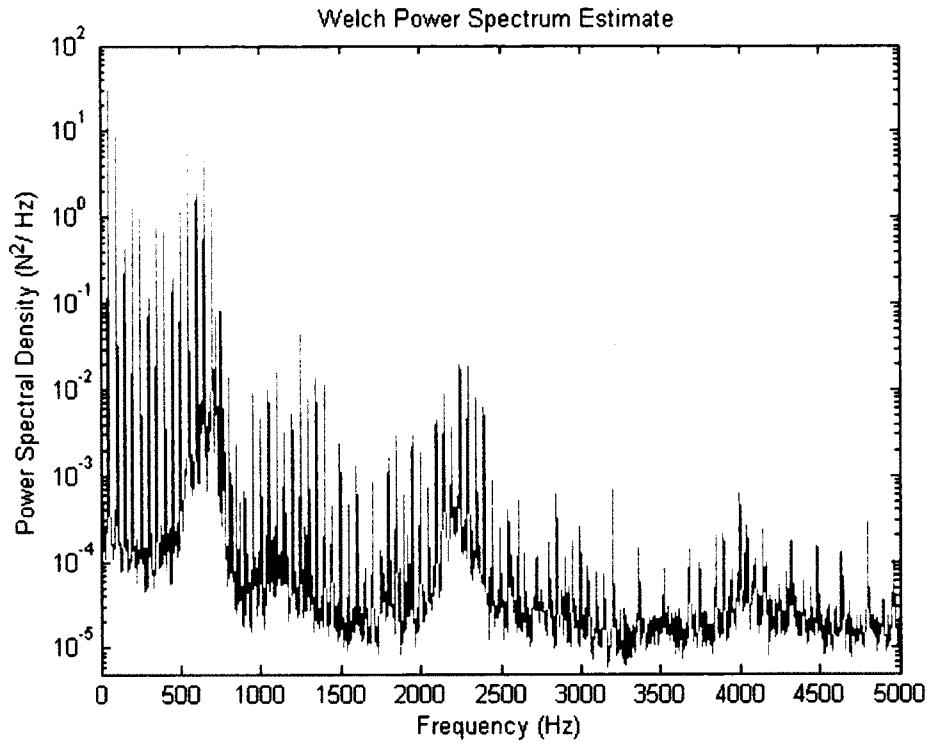


Figure E.2 – Power spectrum of the radial force measurement signal

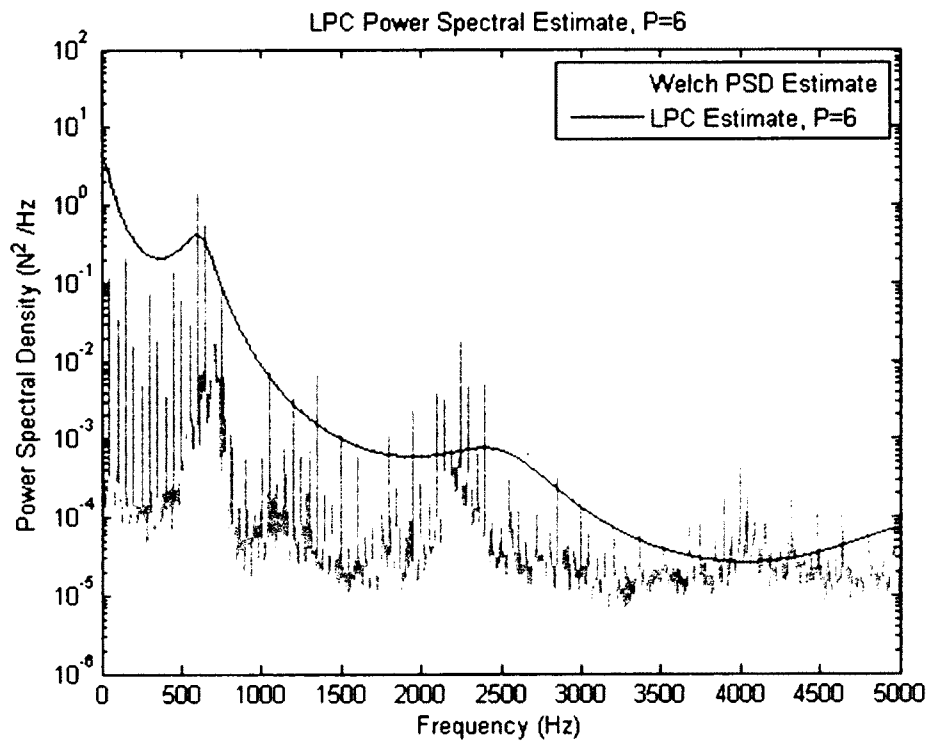


Figure E.3 – LPC spectral estimate, 6th order LPC model

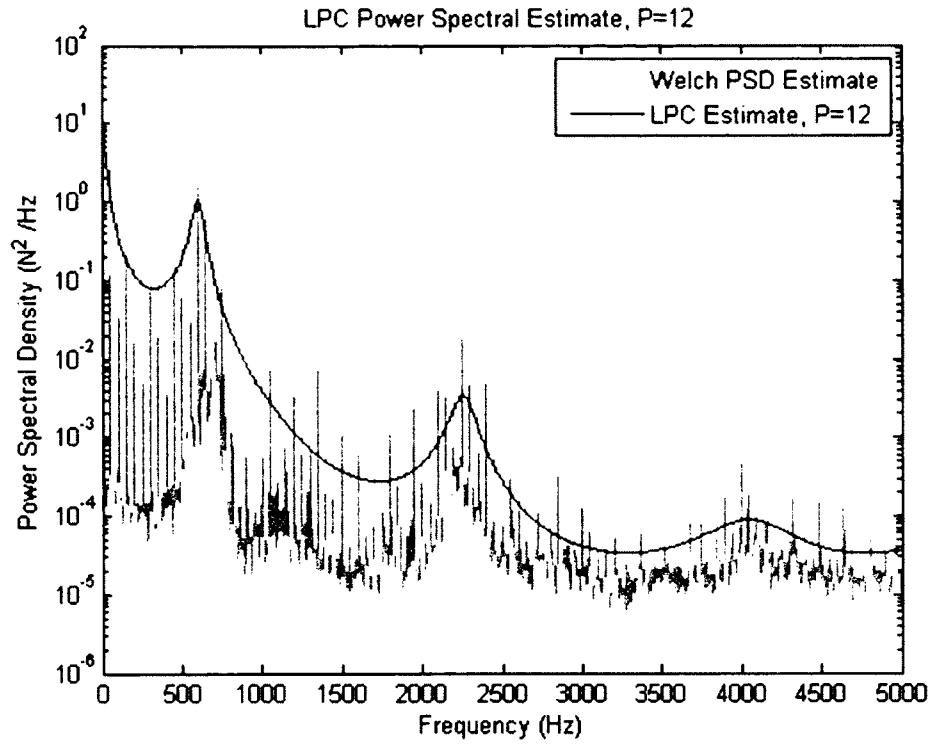


Figure E.4 – LPC spectral estimate, 12th order LPC model

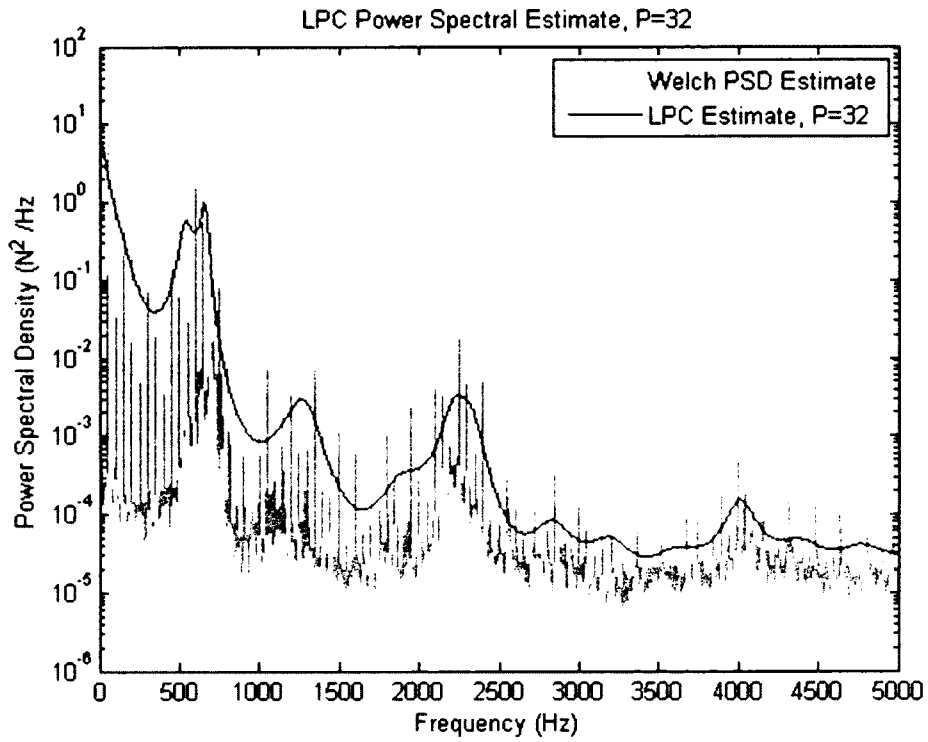


Figure E.5 – LPC spectral estimate, 32nd order LPC model

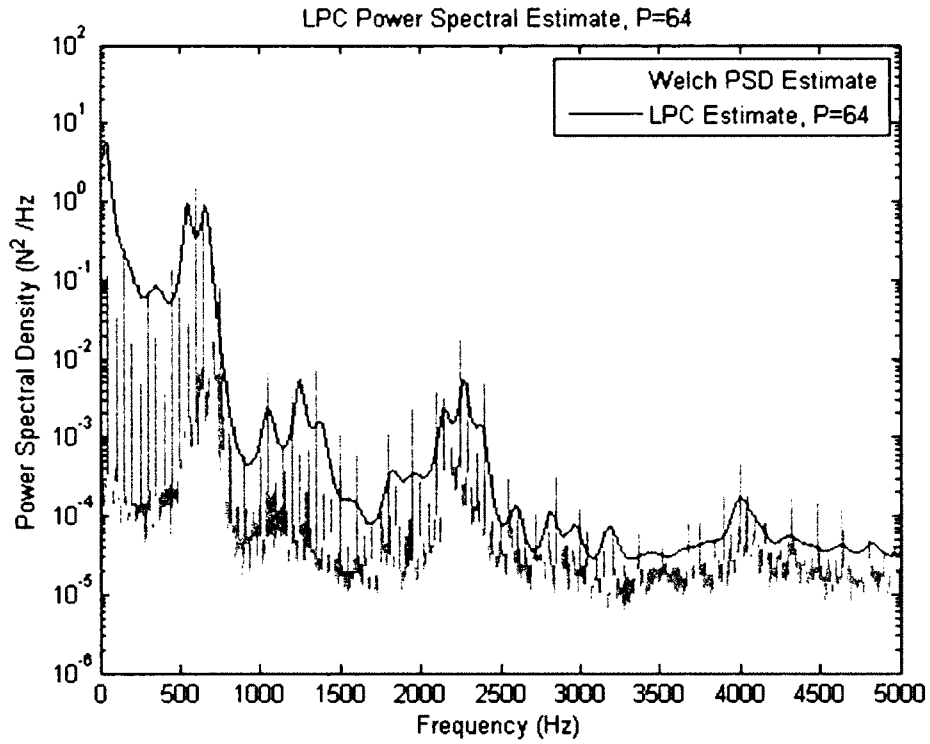


Figure E.6 – LPC spectral estimate, 64th order LPC model

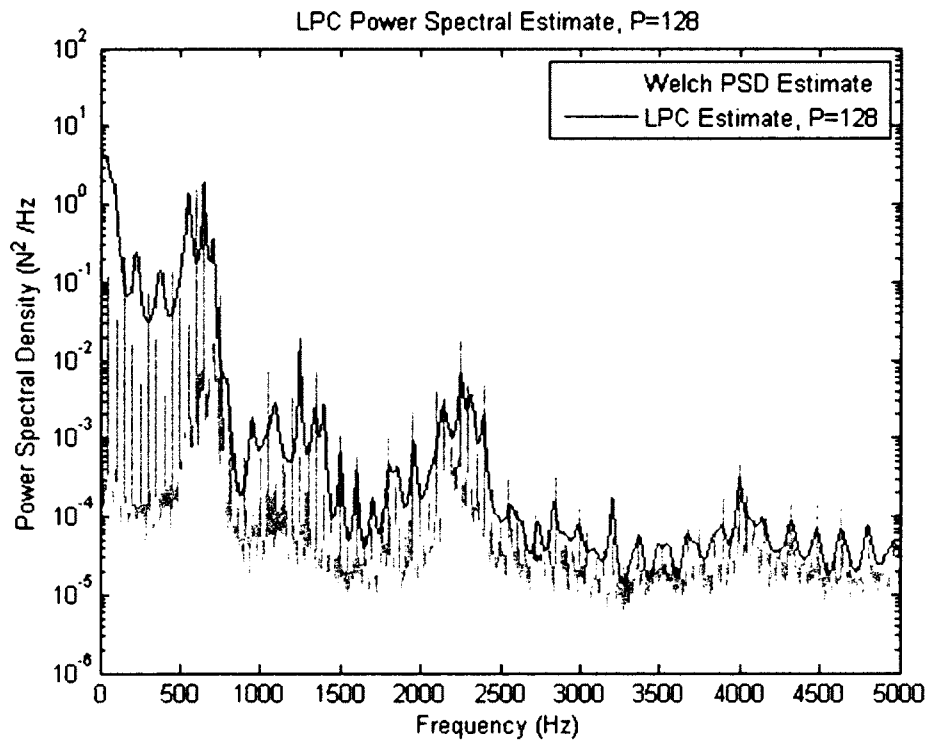


Figure E.7 – LPC spectral estimate, 128th order LPC model

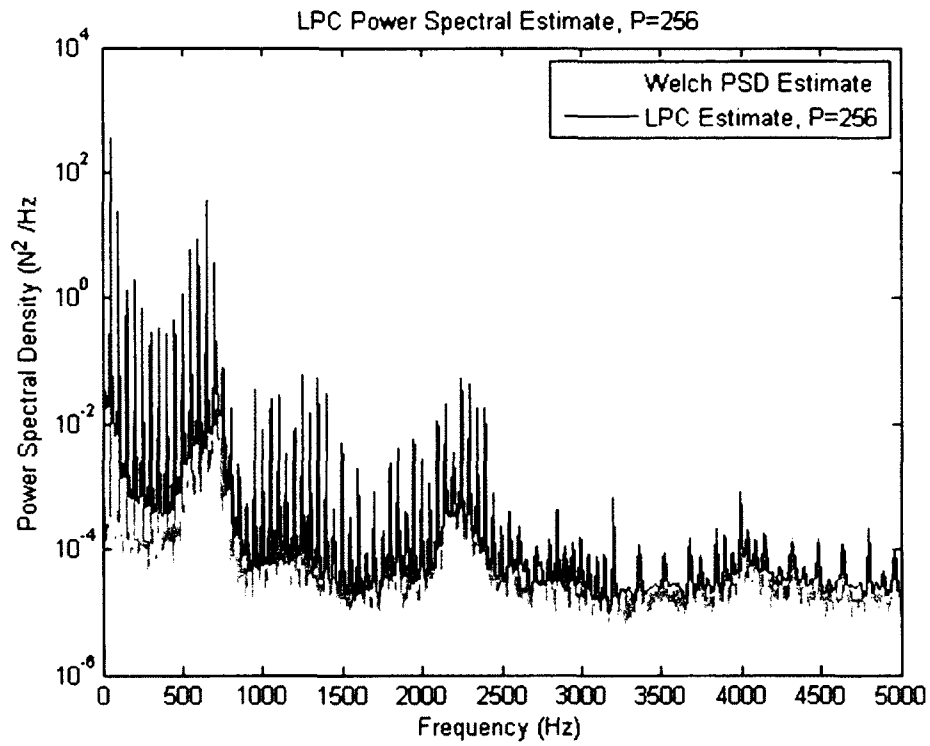


Figure E.8 – LPC spectral estimate, 256th order LPC model

Script to Implement Auto-Regressive Modeling

The built-in LPC function implements forward-predictor lattice estimator. The equations presented in Chapter 6 develop linear prediction using a backwards-prediction lattice estimator. The following script implements linear prediction exactly as defined in this thesis.

```
function [ak varargout] = ARmodel(data,P)
% ARmodel  Auto-Regressive, Bakward-Prediction Model
%
% Usage:   [a v u xhat] = ARmodel(data,P)
%
%          Computes the AR model coefficients

% Error Handling
N = length(data);
if N <= P
    error('Length of data record must be larger than model order');
end

% Initialize Model Vectors
y = zeros(N-P,1);
Y = zeros(N-P,P);

% Build Backward-Prediction AR Data Structure
for row = P:N-1
    y(row-P+1) = data(row+1);
    for col = 1:P
        Y(row-P+1,col) = data(row-col+1);
    end
end

% Least Squares Solution
a = (Y'*Y)\Y'*y;
ak = [1 -a'];

% Process Variance
u = zeros(N-1,1);
xhat = zeros(N-1,1);
for m=1+P:N
    loopsum = 0;
    for k=1:P
        term = a(k)*data(m-k);
        loopsum = loopsum+term;
    end
    u(m) = data(m)-loopsum;
    xhat(m) = loopsum;
end
v = var(u);

if nargout==2
    varargout(1) = {v};
elseif nargout==3
    varargout(1) = {v};
    varargout(2) = {u};
elseif nargout==4
    varargout(1) = {v};
    varargout(2) = {u};
    varargout(3) = {xhat};
end
```

Example Code for LPC-Based Dynamic Parameter Identification

Parametric Methods for Parameter Identification

```
% Get Data (3/4 Immersion upmilling, 3000 rpm, rad, h2)
load Aluminum3k36k4k3QuarterUpRadial.mat
clc

startIdx = 8.58e4;%116525; % Starting index of data sequence
spr = 10240/3000*60; % Samples per revolution
Nrev = 202; % Number of revolutions
stopIdx = round(startIdx + Nrev*spr);
force = (s(startIdx:stopIdx)-505)./1.64*4.448; % N
decay = force(343:430); % Damped vibration profile
N = length(force); % Length of data set
fs = 10240; % Sampling Frequency (Hz)
time = (0:N-1)./fs; % Time vector (sec)
```

Look at How the Dynamics Change

```
% Model Order
P = 6;

% Locate cycles of data
[pks locs] = findpeaks(abs(diff(force)), 'minpeakdistance', 150);

% Compensate for bias
locs = locs(2:end-1)-10;

% Block Out Data
force2 = zeros(Nrev-2, 250);
for i=1:Nrev-2
    force2(i,:) = force((locs(i)-100+1):(locs(i)+150));
end

% Align All Cycles
plot(force2(1,:));
hold on
for i=2:Nrev-2
    [C lags] = xcorr(force2(1,:), force2(i,:), 'coeff');
    [pk idx] = max(C);
    force2(i,:) = circshift(force2(i,:), [0 lags(idx)]);
    plot(force2(i,:));
end

% In the Cut Sections
Fin = force2(:, 39:106);

% Out of Cut Sections
Fout = force2(:, 111:216);

% Detrend The In-Cut Data
[a b] = size(Fin);
Fin2 = zeros(Nrev-2, b);
```

```

for i=1:Nrev-2
    % Define a model Vector
    theta = linspace(pi, (pi+2/3*pi),b);
    X(1,:) = -sin(theta);
    X(2,:) = ones(1,b);
    %X(3,:) = linspace(0,1,65);
    % Multivariate Regression
    out = mvreg(Fin(i,:)',X);
    % Anomaly Time Series
    Fin2(i,:)=out.anom;
end

% Track In-cut Resonances of the Model
Ain = zeros(Nrev-1,P+1);
Zin = zeros(Nrev-1,P);
for i=1:Nrev-2
    [ak g] = lpc(Fin2(i,:),P);           % 6th Order LPC Model
    z = roots(ak);                     % Poles in the z-plane
    Ain(i,:)=ak;
    Zin(i,:)=z;
end

% Track Open Loop Resonances of the Model
Aout = zeros(Nrev-1,P+1);
Zout = zeros(Nrev-1,P);
for i=1:Nrev-2
    [ak g] = lpc(Fout(i,:),P);         % 6th Order LPC Model
    z = roots(ak);                     % Poles in the z-plane
    Aout(i,:)=ak;
    Zout(i,:)=z;
end

% Calculate Formant Frequencies for Each Cycle
flin = zeros(Nrev-2,1);    f2in = zeros(Nrev-2,1);
f3in = zeros(Nrev-2,1);    f4in = zeros(Nrev-2,1);
flout = zeros(Nrev-2,1);   f2out = zeros(Nrev-2,1);
f3out = zeros(Nrev-2,1);   f4out = zeros(Nrev-2,1);
for i=1:Nrev-2
    flin(i)=atan2(imag(Zin(i,1)),real(Zin(i,1)))/2/pi*fs;
    f2in(i)=atan2(imag(Zin(i,3)),real(Zin(i,3)))/2/pi*fs;
    f3in(i)=atan2(imag(Zin(i,5)),real(Zin(i,5)))/2/pi*fs;
    %f4in(i)=atan2(imag(Zin(i,7)),real(Zin(i,7)))/2/pi*fs;
    flout(i)=atan2(imag(Zout(i,1)),real(Zout(i,1)))/2/pi*fs;
    f2out(i)=atan2(imag(Zout(i,3)),real(Zout(i,3)))/2/pi*fs;
    f3out(i)=atan2(imag(Zout(i,5)),real(Zout(i,5)))/2/pi*fs;
    %f4out(i)=atan2(imag(Zout(i,7)),real(Zout(i,7)))/2/pi*fs;
end

```

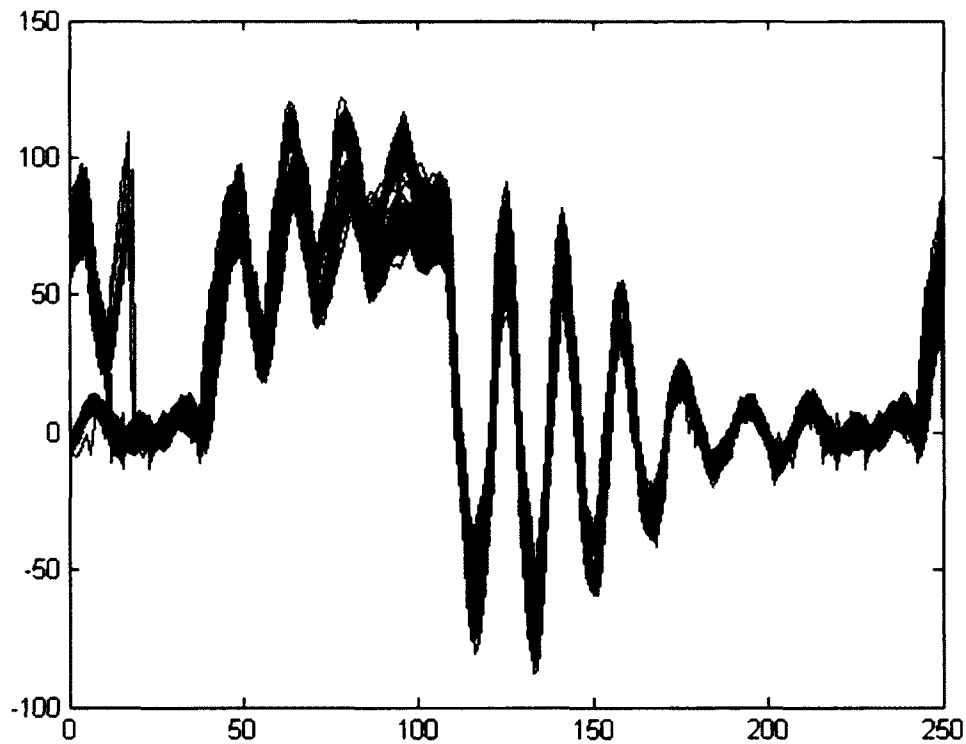



Figure E.9 - Overlay of force profiles aligned by the cross-correlation function

Sort States

```

fin = sort(abs([f1in f2in f3in]),2);
f1in = fin(:,1);
f2in = fin(:,2);
f3in = fin(:,3);
fout = sort(abs([f1out f2out f3out]),2);
f1out = fout(:,1);
f2out = fout(:,2);
f3out = fout(:,3);

```

Visualizing the System Description

```

% Find Poles Around 3rd Mode, Exclude Outliers
f3inID = zeros(Nrev-2,1);
f3outID = zeros(Nrev-2,1);
for i=1:Nrev-2
    if f3in(i) <= 5000
        f3inID(i) = 1;
    end
    if f3out(i) <= 5000
        f3outID(i) = 1;
    end
end
end

```

```

% Average Pole Locations
Ain2 = zeros(sum(f3inID),7);
Aout2 = zeros(sum(f3outID),7);
n=1; m=1;
for i=1:Nrev-2
    if f3inID(i)==1
        Ain2(n,:)=Ain(i,:);
        n=n+1;
    elseif f3outID(i)==1
        Aout2(m,:)=Aout(i,:);
        m=m+1;
    end
end
akCL = mean(Ain);
zCL = roots(akCL);           % Poles in the z-plane
akOL = mean(Aout);
zOL = roots(akOL);         % Poles in the z-plane

figure(2)
subplot(2,2,1:2)
hold off
[hOL f] = freqz(1,akOL,4096,'half',fs);
[hCL f] = freqz(1,akCL,4096,'half',fs);
plot(f,20*log10(abs(hOL)),'b')
hold on
plot(f,20*log10(abs(hCL)),'r')
title('Frequency Response of the LPC Model')
xlabel('Magnitude (dB)')
ylabel('Frequency (Hz)')
legend('Opel Loop','Closed Loop')
subplot(2,2,3)
hold off
zplane([],zOL)
title('OL Poles of the LPC Model');
subplot(2,2,4)
hold off
zplane([],zCL)
title('CL Poles of the LPC Model');

```

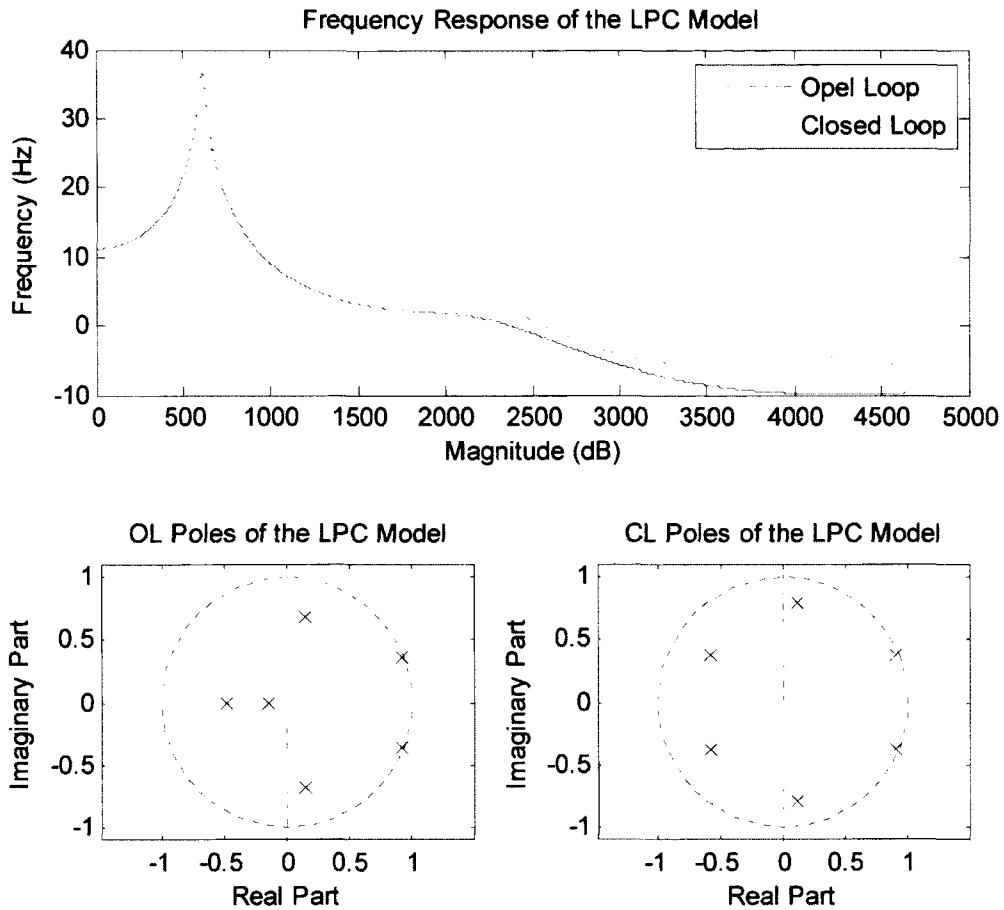


Figure E.10 – Average resonance as determined by LPC

Make a Movie

```

Frame(Nrev-1) = struct('cdata', [], 'colormap', []);
g=figure(3);
set(g, 'Position', [100 100 800 800]);
n=1;
for j=1:Nrev-2
    akOL = Aout(j,:);
    zOL = roots(akOL);
    akCL = Ain(j,:);
    zCL = roots(akCL);

    subplot(3,2,1:2)
    hold off
    step = 0;
    for k=1
        t = time((locs(j)-80+1+step):(locs(j)+633+step));
        plot(t,force((locs(j)-80+1+step):(locs(j)+633+step)), 'k')
        hold on
        plot([t(120) t(120)], [-100 110], 'r')
    
```

```

        title(strcat('Measured Force - 3/4 Upmilling at 3000 RPM,
Cycle =',int2str(j)))
        xlabel('Time (sec)')
        ylabel('Force (N)')
        xlim([min(t) max(t)])
        ylim([-100 110])
        step = step+35;
        Frame(n) = getframe(gcf);
        n = n+1;
    end
    subplot(3,2,3:4)
    hold off
    [hOL f] = freqz(1,akOL,4096,'half',fs);
    [hCL f] = freqz(1,akCL,4096,'half',fs);
    plot(f,20*log10(abs(hOL)),'b')
    hold on
    plot(f,20*log10(abs(hCL)),'r')
    title(strcat('Frequency Response of the LPC Model, Cycle
=',int2str(j)))
    ylabel('Magnitude (dB)')
    xlabel('Frequency (Hz)')
    legend('Open Loop','Closed Loop')
    xlim([0 5120])
    ylim([-10 40])
    grid on
    subplot(3,2,5)
    hold off
    zplane([],zOL)
    title('OL Poles of the LPC Model');
    subplot(3,2,6)
    hold off
    zplane([],zCL)
    title('CL Poles of the LPC Model');
    subplot(3,2,1:2)
    for k=1:5
        hold off
        t = time((locs(j)-80+1+step):(locs(j)+633+step));
        plot(t,force((locs(j)-80+1+step):(locs(j)+633+step)),'k')
        hold on
        plot([t(120) t(120)],[-100 110],'r')
        title(strcat('Measured Force - 3/4 Upmilling at 3000 RPM,
Cycle =',int2str(j)))
        xlabel('Time (sec)')
        ylabel('Force (N)')
        xlim([min(t) max(t)])
        ylim([-100 110])
        step = step+35;
        Frame(n) = getframe(gcf);
        n = n+1;
    end
    Frame(n) = getframe(gcf);
    n=n+1;
end

% Play the Movie
movie(gcf,Frame,1,5)

```

Plotting for Formant Frequency Tracking

```
figure(4)
hold off
subplot(2,1,1)
    plot(1:Nrev-2,flout, '.', 'color',0.6.*[1 1 1])
    hold on
    plot(1:Nrev-2,f2out,'k*')
    plot(1:Nrev-2,f3out,'ko')
    ylim([0 5000])
    title('Open Loop Resonant Frequencies')
    xlabel('Spindle Revolution (cycle)')
    ylabel('Frequency (Hz)')
    legend('First Mode','Second Mode','Third
Mode','location','northwest')
subplot(2,1,2)
    plot(1:Nrev-2,flin, '.', 'color',0.6.*[1 1 1])
    hold on
    plot(1:Nrev-2,f2in,'k*')
    plot(1:Nrev-2,f3in,'ko')
    ylim([0 5000])
    title('Closed Loop Resonant Frequencies')
    xlabel('Spindle Revolution (cycle)')
    ylabel('Frequency (Hz)')
    legend('First Mode','Second Mode','Third
Mode','location','northwest')

figure(5)
hold off
    plot(1:Nrev-2,flout,'o','color',[0 0 1],'markersize',4.5)
    hold on
    plot(1:Nrev-2,flin, '.', 'color',[0 .7 0])
    plot(1:Nrev-2,f2out,'square','color',[1 0 0],'markersize',4)
    plot(1:Nrev-2,f2in,'*', 'color',[.7 0 .7])
    plot(1:Nrev-2,f3out,'^','color',[0.9 0.8 0],'markersize',5)
    plot(1:Nrev-2,f3in,'+', 'color',[.4 .7 .7],'markersize',5)
    ylim([0 5000])
    title('Comparison of Open-Loop & Closed-Loop Resonant Frequencies')
    xlabel('Spindle Revolution (cycle)')
    ylabel('Frequency (Hz)')
    legend('OL Mode 1','CL Mode 1','OL Mode 2','CL Mode 2',...
        'OL Mode 3','CL Mode 3','location','southwest')
```

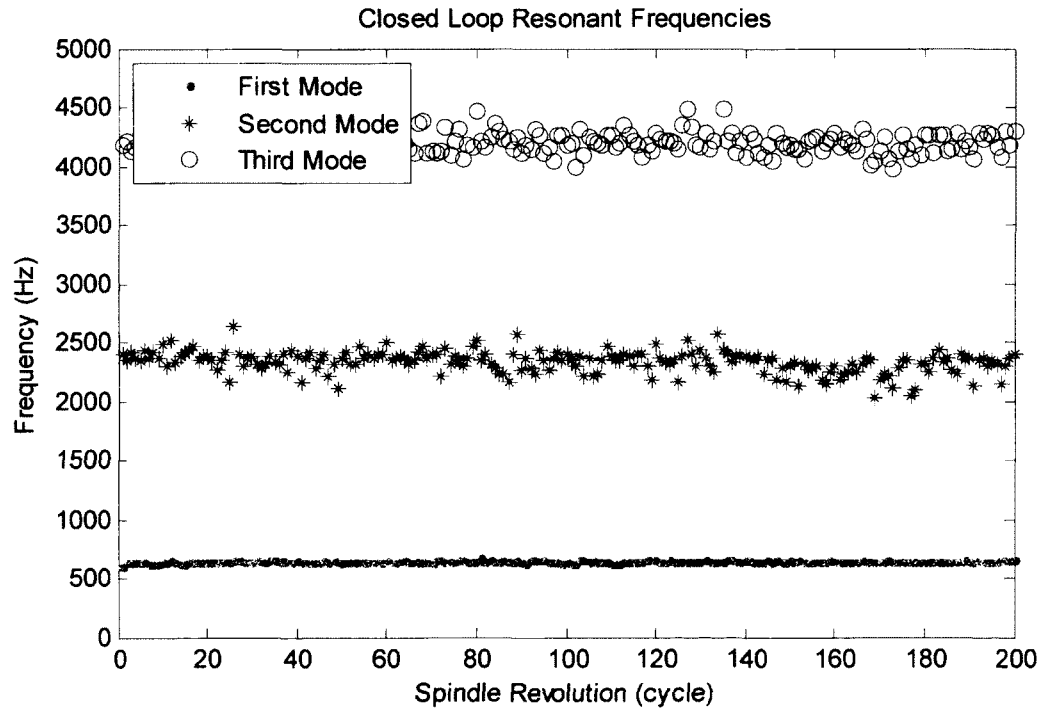
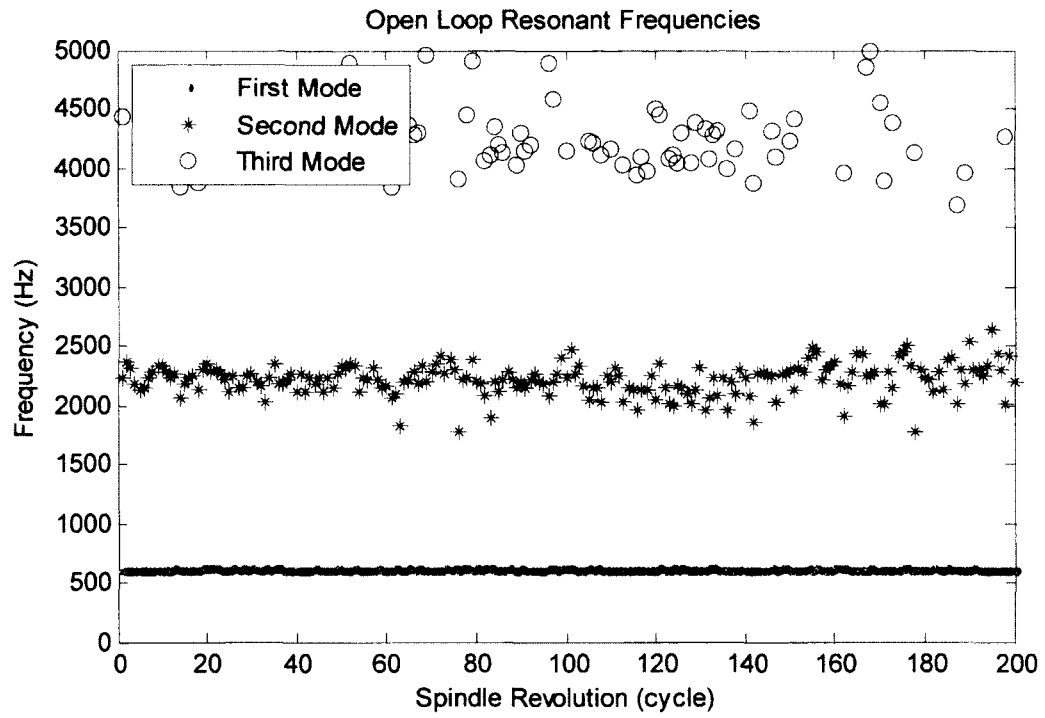


Figure E.11 - Open loop and closed loop resonance as determined by LPC

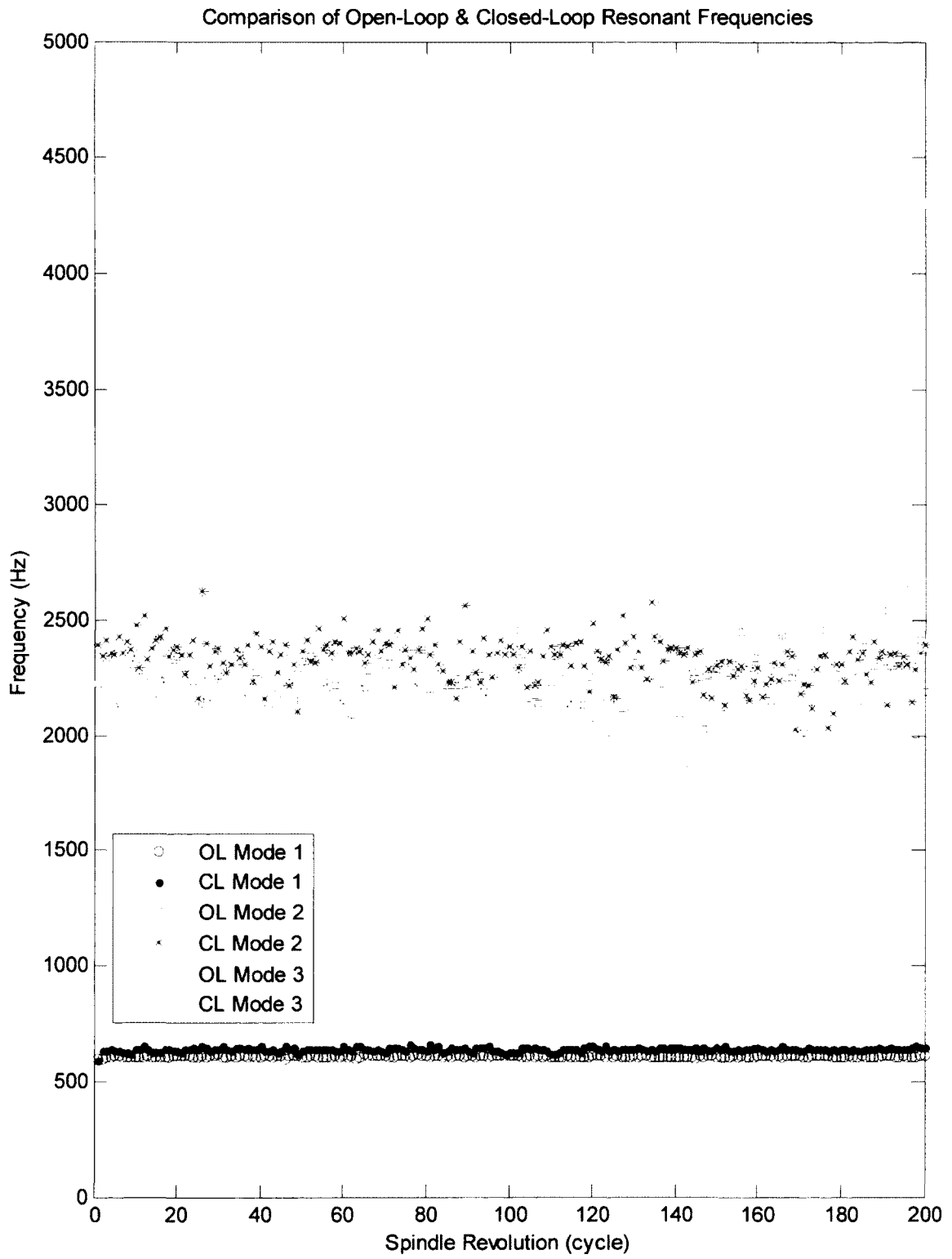


Figure E.12 - Tracking resonance with LPC

Example Test for Whiteness

```
% In-cut profile
x = Fout(100,:);

% Estimate
P=6;
[a v e xhat] = ARmodel(x,P);
m = 0:length(x)-1;

% Autocorrelation
[C lags] = xcorr(e(P+1:end),e(P+1:end),'coeff');

% Plotting
figure(1)
subplot(3,1,1)
    hold off
    plot(m,x,'color',[0.3 0.3 0.3])
    hold on
    plot(m(P+1:end),xhat(P+1:end),'kx');
    xlim([0 100])
    ylim([-100 200])
    legend('Measurement','Linear Prediction
Model','location','northeast')
    title('Example of 6t Order Linear Prediction Model')
    xlabel('Data Index')
    ylabel('Force (N)')

subplot(3,1,2)
    hold off
    plot(m(P+1:end),e(P+1:end),'k')
    xlim([0 100])
    ylim([-10 10])
    title('Residual White Noise Error Sequence')
    xlabel('Data Index')
    ylabel('Force (N)')
    grid on

subplot(3,1,3)
    hold off
    stem(lags,C,'k*')
    title('Autocorrelation of the Error Sequence')
    xlabel('Lag Index')
    ylabel('Correlation Coefficient')
    xlim([-50 50])
    ylim([-1 1])
    grid on
```

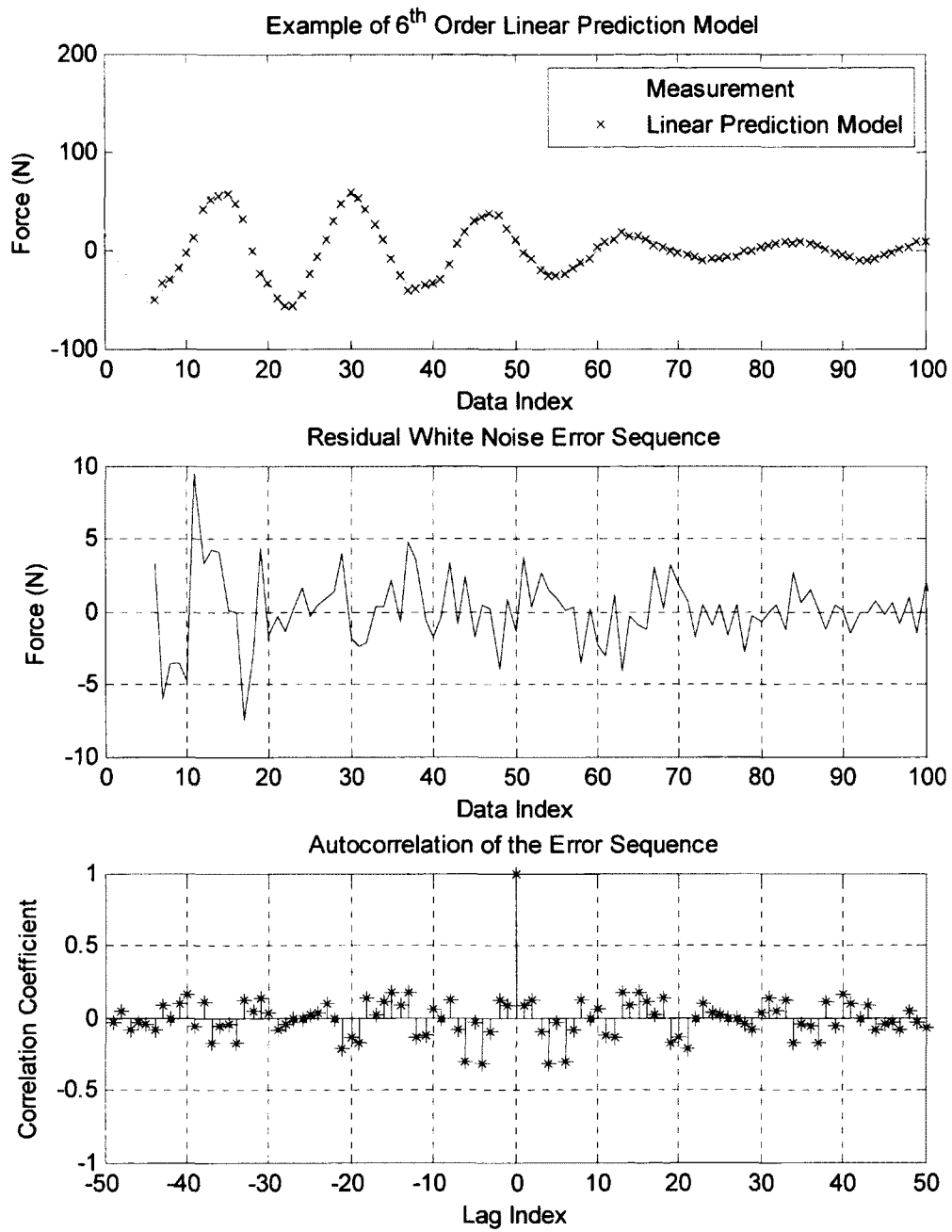



Figure E.13 – Test for whiteness by using the autocorrelation

Example Code for Implementing the Weiner Filter

Measured Data to Filter

```
% Strain Signal to Filter
load radial3000h2.mat;
force = (s-506)./1.64*4.448; % N
N = length(force);
time = (0:N-1)./10240;
```

Static Model used for Training Signal

% Example, 3000 rpm Upmilling

```
% Parameters
rpm=2990; % input spindle speed
fs=10240; % sampling frequency
numpercycle=fs*60/rpm; % number of data points per rotation
AD = 1; %0.125*25.4;
Feed=8.378; % assign the feedrate
ft=Feed*25.4/rpm/N; % feed per tooth : mm/tooth
degree=1/fs*rpm*360/60; % angular increment for simulation

% Upmilling Geometry for Selected Radial Immersion
immersion = '3quarter';
switch(lower(immersion))
    case {'quarter'}
        enter=pi;
        exit=pi+1/3*pi;
        maxphi = exit;
    case {'half'}
        enter=pi;
        exit=pi+1/2*pi;
        maxphi = pi/2;
    case {'3quarter'}
        enter=pi;
        exit=pi+2/3*pi;
        maxphi = pi/2;
end

% Build the Static Model
Ktcft=72; % unit: N/mm^2 Multiplied by feedrate
Kte=20; % unit: N/mm

alpha=0; % initial value of reference locating angle
for j=1:numpercycle
    theta=(alpha+degree*(j-1))/180*pi;
    if (theta<=exit) && (theta>=enter)
        Ft(j)=(-Ktc*sin(theta)+Kte)*AD;
    else
        Ft(j)=0;
    end
end
cycles=25;
model = [];
```

```

    for n = 1:cycles
        model = vertcat(model,Ft');
    end

    % Rotate and Trim Model
    model = circshift(model(1:N),[-53 0]);

```

Develop Weiner Block-Adaptive Filter

```

% FIR Model Order
P = 7;

% Build the Observation Matrix, Y
y = force;
x = model(P:end)';
Y = zeros(N-P+1,P);
for i = 1:N-P+1;
    Y(i,:) = fliplr(y((i):(i+P-1)));
end

% FIR Least Squares Weighting Factors
w = (Y'*Y)\Y'*x';

```

Apply the Filter to the Measured Signal

```

% Filtering
x1h = filter(w,1,force);

```

Plotting

```

% Estimate Plotting
figure(1)
xmax = 100;
hold off
plot((0:N-1)/10240*1000,force,'-','color',[0.65 0.65 0.65])
hold on
plot((0:N-1)/10240*1000,model,'-','color',[0.7 0
0.7],'linewidth',1)
title('Static Profile Used to "Train" the Weiner Filter')
xlabel('Time (ms)')
ylabel('Force, (N)')
axis([0 xmax -100 120])
legend('Measurement','Static Training
Signal','location','southwest')

figure(2)
subplot(2,1,1)
hold off
plot((0:N-1)/10240*1000,force,'-','color',[0.65 0.65 0.65])
hold on
plot((0:N-1)/10240*1000,x1h,'-','color',[0.7 0
0.7],'linewidth',1)
title('Static Signal Estimate')
xlabel('Time (ms)')

```

```

        ylabel('Force, (N)')
        axis([0 xmax -100 120])
        legend('Measurement','Weiner Estimate','location','southwest')

subplot(2,1,2)
    hold off
    plot((0:N-1)/10240*1000,force,'-','color',[0.65 0.65 0.65])
    hold on
    nMA = 4;
    bMA = ones(1,nMA)./nMA;
    plot((0:N-1)/10240*1000,filter(bMA,1,x1h),'-','color',[0.7 0
0.7],'linewidth',1)
    title('Static Signal Estimate - Smoothed with MA Filter, P=4')
    xlabel('Time (ms)')
    ylabel('Force, (N)')
    axis([0 xmax -100 120])
    legend('Measurement','Smoothed Weiner
Estimate','location','southwest')

% Poles and Zeros
figure(4)
    hold off
    zplane(roots(w),[]);
    title('Zeros of the Weiner Filter')

% Frequency Response
figure(5)
    [H, F] = freqz(w,1,512,'half',10240);
    subplot(2,1,1)
        plot(F,20*log10(abs(H)));
        title('Frequency Response of the Weiner Filter')
        xlabel('Frequency (Hz)')
        ylabel('Amplitude Ratio (dB)')
        xlim([0 5000])
        ylim([-30 20])
        grid on
    subplot(2,1,2)
        plot(F,atan2(imag(H),real(H))*180/pi);
        xlabel('Frequency (Hz)')
        ylabel('Phase (deg)')
        xlim([0 5000])
        ylim([-90 180])
        set(gca,'ytick',-90:45:180)
        grid on

```

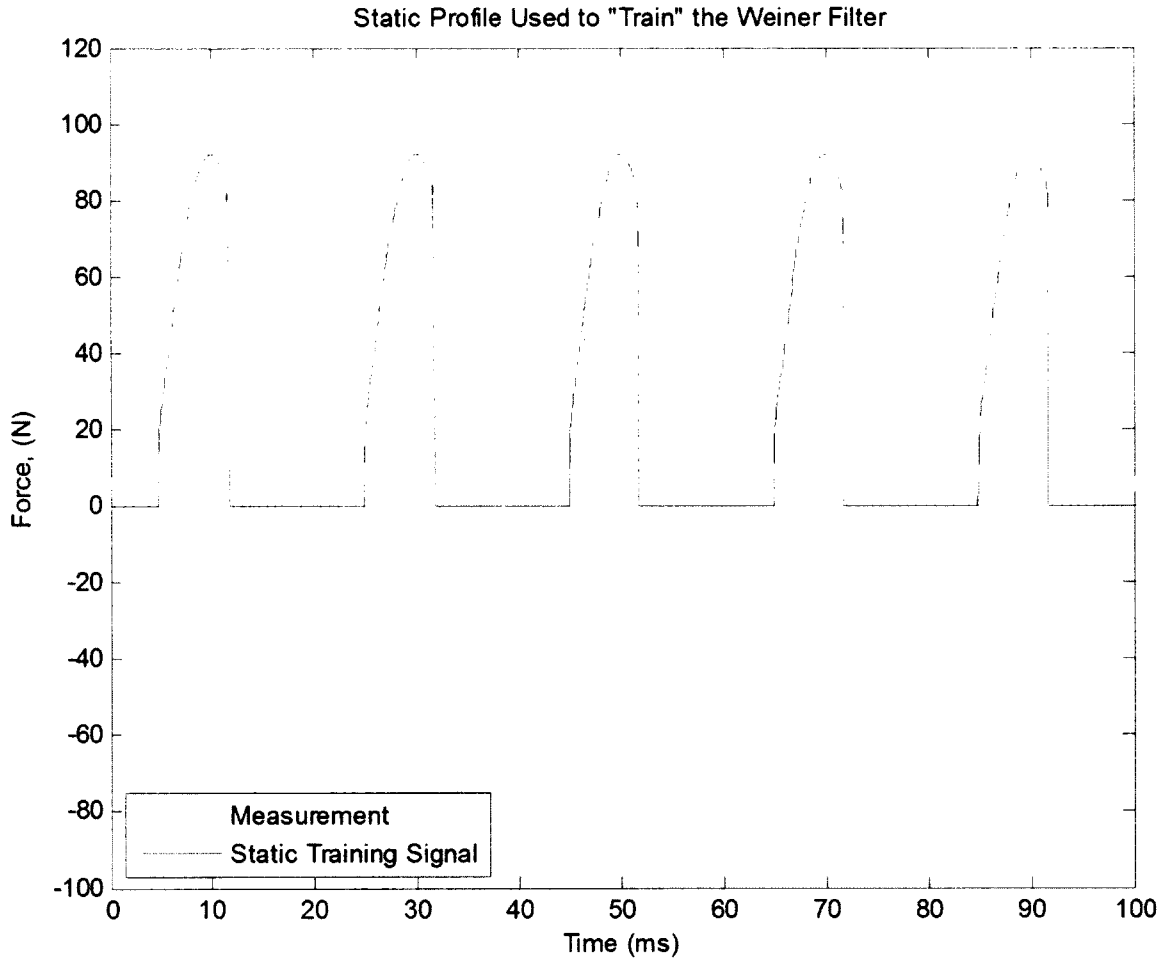


Figure E.14 - Static model used to train the Weiner filer

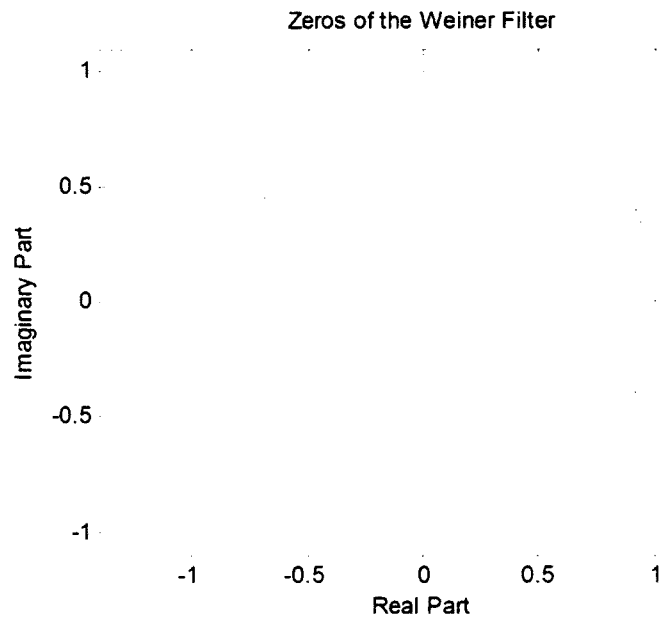


Figure E.15 - Zeros of the Weiner Filter

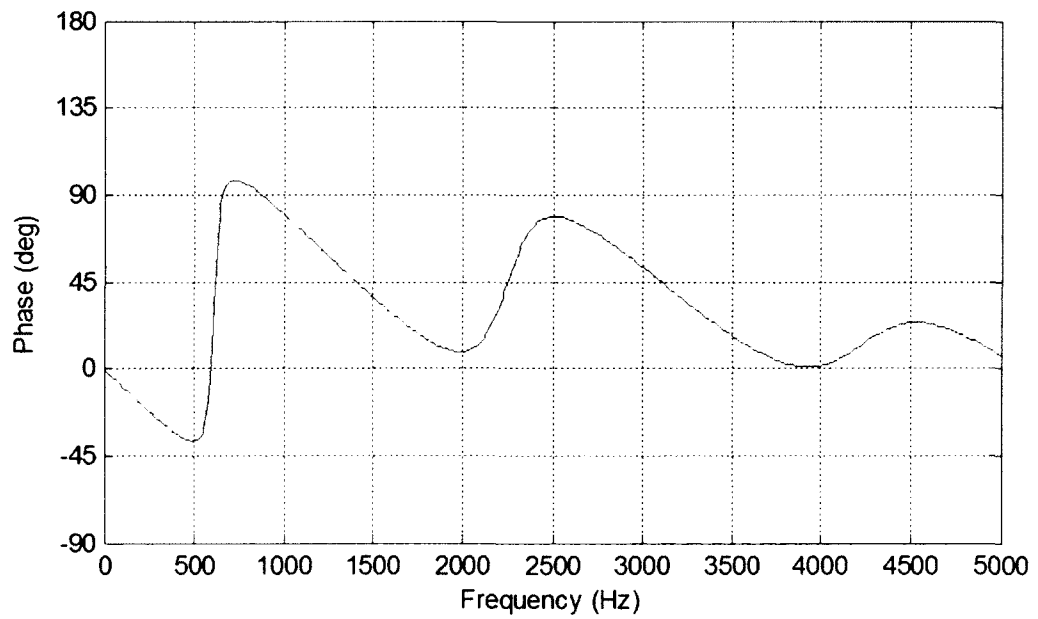
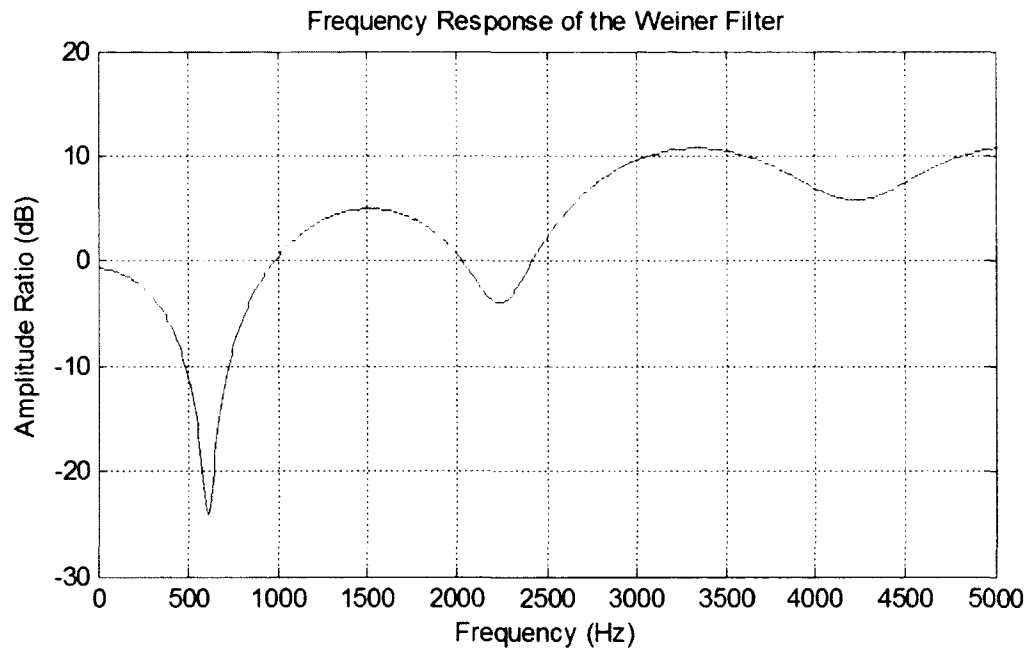


Figure E.16 - Frequency response of the Weiner filter

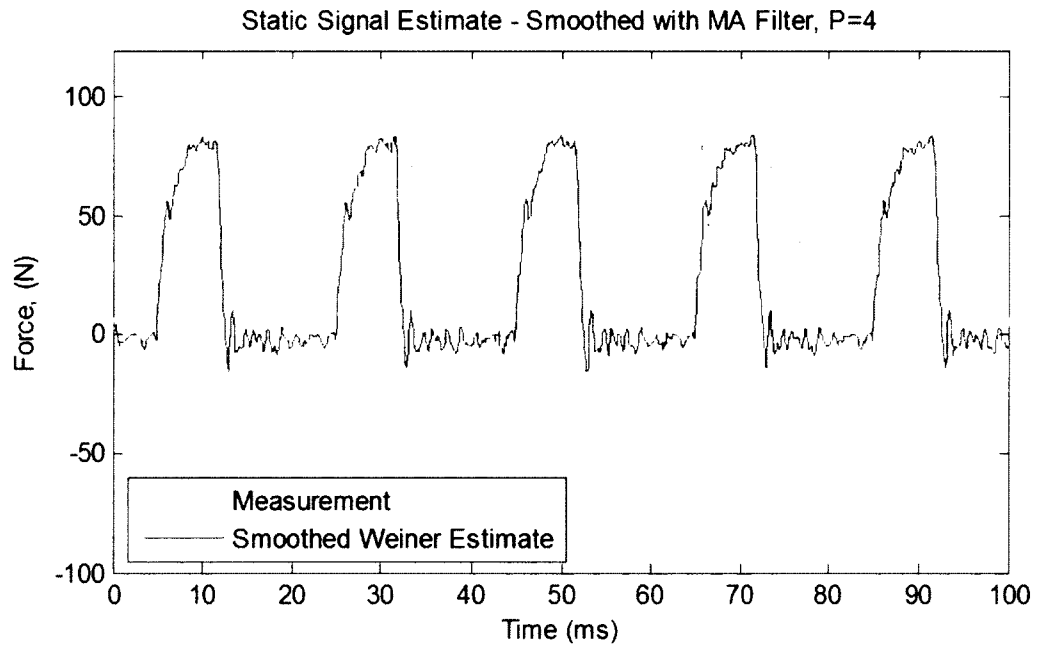
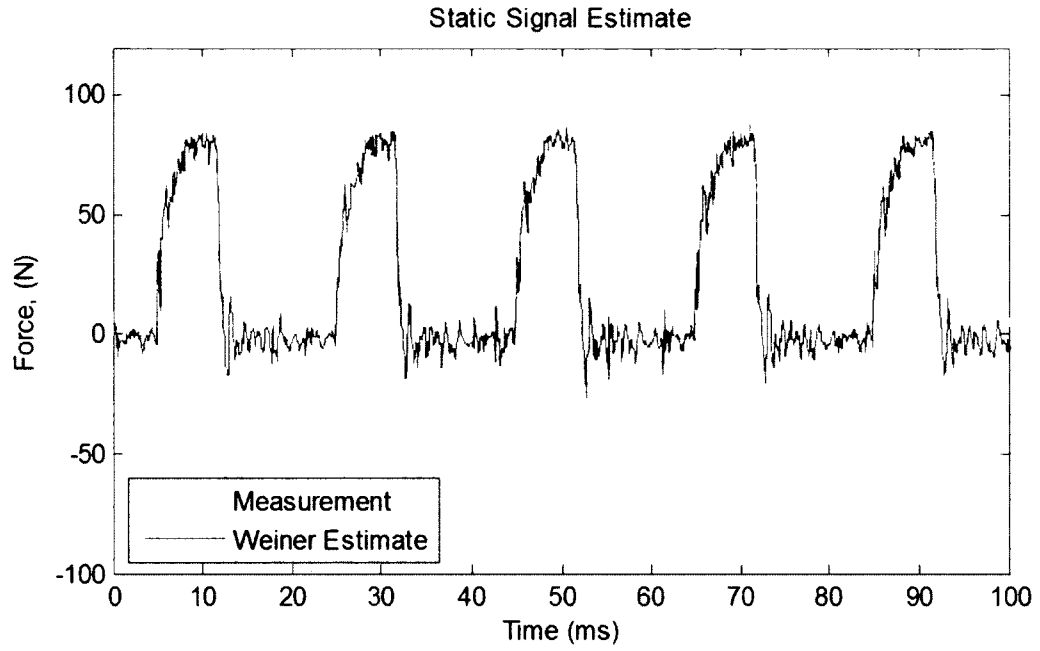


Figure E.17 - Implementation of the Wiener filter

Example Code for Implementing the Harmonic Kalman Filter

Static Force Profile

% Example, 3000 rpm Upmilling

```
% Parameters
rpm=3000;           % input spindle speed
N=1;               % number of cutting tooth
fs=10000;          % sampling frequency
numpercycle=fs*60/rpm; % number of data points per rotation
AD = 1; %0.125*25.4;
Feed=8.378;        % assign the feedrate
ft=Feed*25.4/rpm/N; % feed per tooth : mm/tooth
degree=1/fs*rpm*360/60; % angular increment for simulation

% Upmilling Geometry for Selected Radial Immersion
immersion = '3quarter';
switch(lower(immersion))
    case {'quarter'}
        enter=pi;
        exit=pi+1/3*pi;
        maxphi = exit;
    case {'half'}
        enter=pi;
        exit=pi+1/2*pi;
        maxphi = pi/2;
    case {'3quarter'}
        enter=pi;
        exit=pi+2/3*pi;
        maxphi = pi/2;
end

% Build the Static Force Model
Ktc=95;           % unit: N/mm^2 % Multiplied by feedrate
Kte=0;           % unit: N/mm

alpha=0;         % initial value of reference locating angle
for j=1:numpercycle
    theta=(alpha+degree*(j-1))/180*pi;
    if (theta<=exit) && (theta>=enter)
        Ft(j)=(-Ktc*sin(theta)+Kte)*AD;
    else
        Ft(j)=0;
    end
end
cycles=20;
model = [];
for n = 1:cycles
    model = vertcat(model,Ft');
end
```


DFS of Static Force Profile

```
% Spectrum of the Static Profile
N = length(model);
m = 0:N-1;
t = m./fs;
f = m/N*fs;
DFS = 2.*abs(fft(model-mean(model)))./length(model); % One-sided
spectrum amplitudes

% Power Spectrum (Periodogram Definition)
Power = DFS.^2./length(model)./fs;
Power2 = Power(1); % [380 38 1]; % Informed from LPC Model

% Identify Non-Zero Modes
ck = DFS(21:20:N);
Pxx = Power(21:20:N);
Pyy = Power2;
n = length(ck);
```

Harmonic Kalman Filter Model

```
% Model Order Selection
p = 10; % Number of Harmonics to Model
q = 1; % Number of Resonant Modes to Model
s = 2*(p+q); % Total Number of States

% Frequency Vector for Harmonics, Resonance, Combined
w1 = 2.*pi.*f(21:20:p*21);
w2 = 2.*pi.*[650;% 2170 4200];
w = [w1 w2];

% Building the A Matrix
A = zeros(s);
for i=1:s
    for j=1:s
        if rem(i,2)~=0 && j==(i+1)
            A(i,j)=1;
        end
        if rem(i,2)==0 && i==(j+1)
            A(i,j) = -w(i/2)^2;
        end
    end
end

% Building the B Matrix
B = zeros(s,p+q);
count=1;
for i=1:s
    if rem(i,2)==0
        B(i,count) = 1;
        count=count+1;
    end
end
```

```

% Building the Observation Matrix
H = zeros(1,s);
count=1;
for j=1:2*p
    if rem(j,2)~=0
        H(j) = abs(ck(count))^2;
        count=count+1;
    end
end

% Initialize D Matrix
D = zeros(1,p+q);

% State Space Model
sys = ss(A,B,H,D);
% sysr = minreal(sys);

```

Kalman Gain Computation

```

% Parameters
Qn = diag([ck(1:p)' ck(1)].^2); % Driving Process Noise
Rn = 1.2; % N^2 % Variance of the additive noise

% Discrete Gain Estimator for Continuous Plant
[kest,L,P] = kalman(sys,Qn,Rn);

```

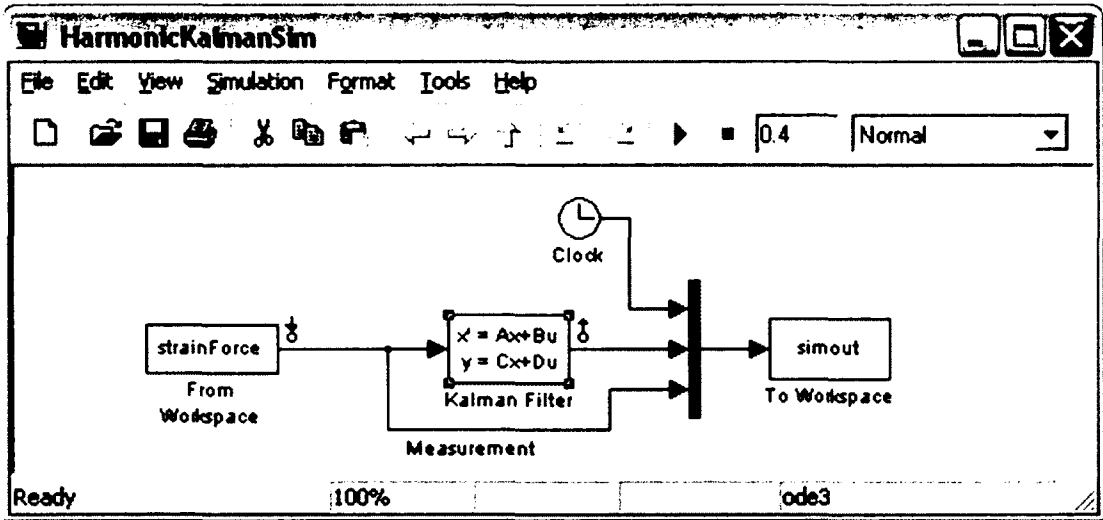
Filter Implementation

```

% Strain Signal to Filter
load radial3000h2.mat;
force = (s2-506)./1.64*4.448; % N
N2 = length(force);
time = (0:N2-1)./10240;

% Simulate the Model
T = 1/10240;
strainForce = [time', force'];
sim('HarmonicKalmanSim.mdl',1);

```



Function Block Parameters: Kalman Filter

State Space

State-space model:
 $\dot{x} = Ax + Bu$
 $y = Cx + Du$

Parameters

A:

B:

L:

C:

H:

D:

Initial conditions:

Absolute tolerance:
 auto

State Name: (e.g., 'position')
 "

OK Cancel Help Apply

Figure E.18 - Simulink model for the harmonic Kalman filter

Static Force Profile and DFS Plotting

```
figure(1)
subplot(2,1,1)
    hold off
    plot(t.*1000,model,'b.--')
    title('Static Model')
    xlabel('Time (ms)')
    ylabel('Force, (N)')
    xlim([0 205])
    ylim([0 120])
subplot(2,1,2)
    stem(f,DFS,'b.')
    xlim([0 2000])
    title('Discrete Time Fourier Series')
    ylabel('Fourier Series Coefficient')
    xlabel('Frequency (Hz)')
figure(2)
    hold off
    plot(1:n,2*cumtrapz(1:n,abs(ck).^2)./trapz(1:n,abs(ck).^2).*100,
'bo:')
    xlim([0 30])
    ylim([0 100])
    grid on
    title('Kalman Filter: Model Order Selection')
    xlabel('Number of Hamonics')
    ylabel('Percent of Signal Energy')
```

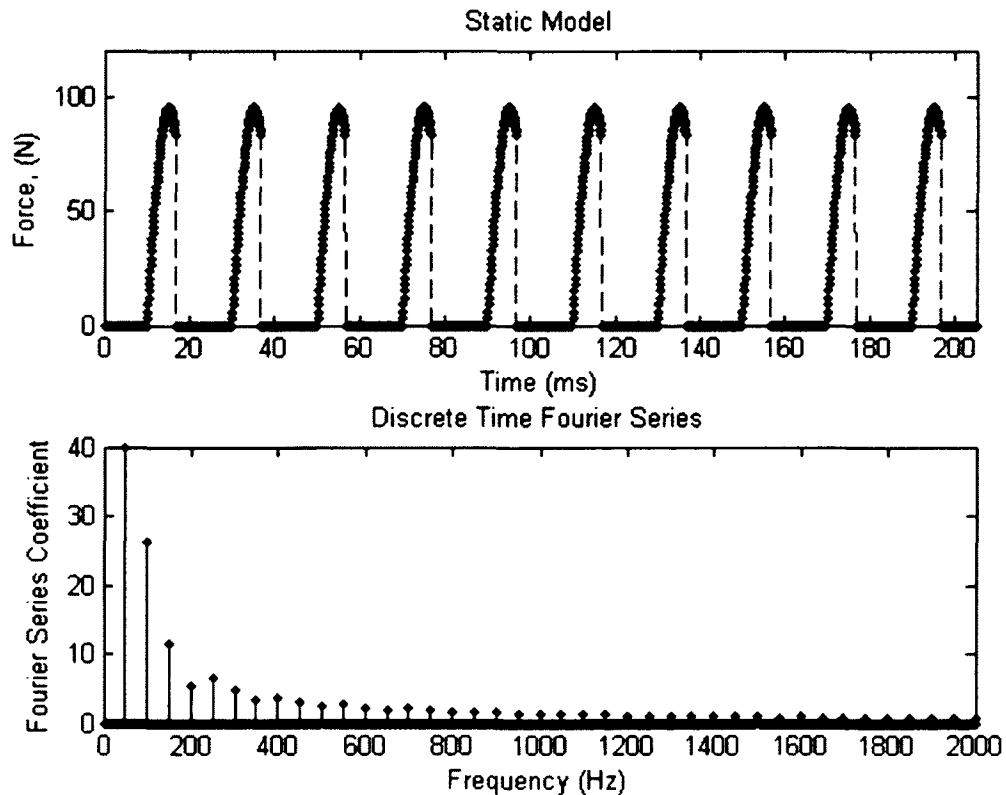


Figure E.19 - Static force profile and DFS

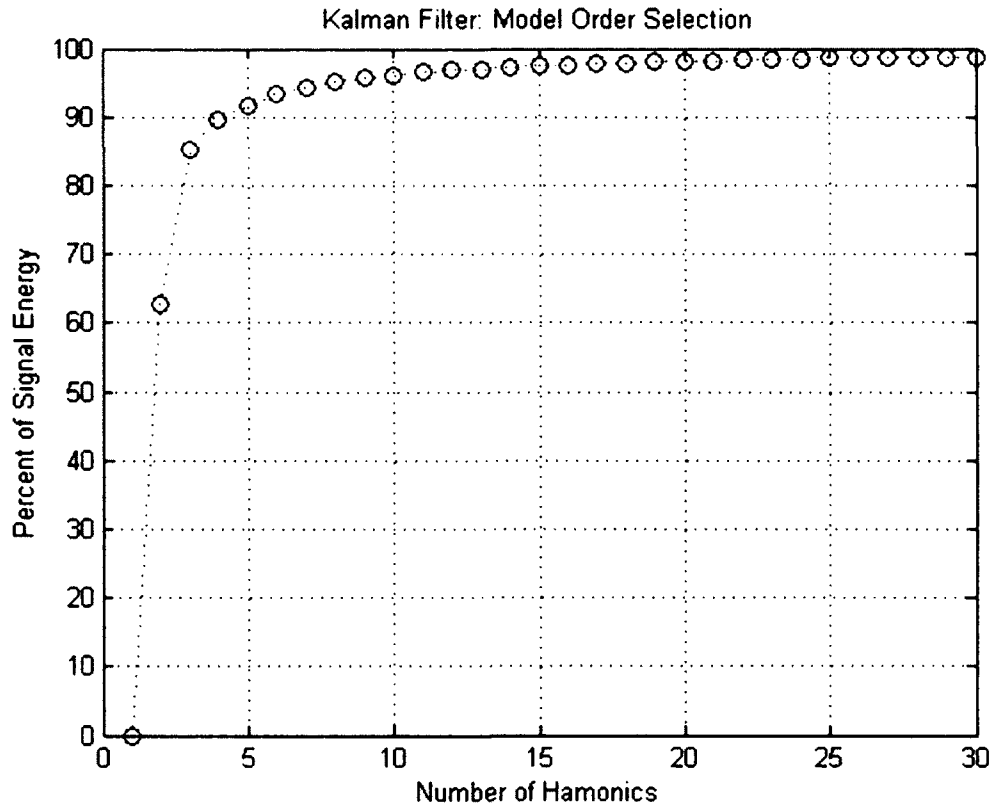


Figure E.20 - Integration of the static model power spectrum

Kalman Filter Plotting

```

figure(3)
    hold off
    plot((simout(:,1)-.61)*1000,simout(:,3),'color',[0.65 0.65
0.65]);
    hold on
    plot((simout(:,1)-
.61)*1000,simout(:,2)+mean(model)+1,'color',[0.7 0 0.7],'linewidth',1);
    legend('Measurement','Kalman Estimate','location','southwest')
    title('Harmonic Kalman Filter')
    xlabel('Time, (ms)')
    ylabel('Force (N)')
    xlim([0 60])
    ylim([-100 120])

```

```

figure(4)
    load KalmanBode.mat
    [MAG PHASE W] = bode(Model);
    for count = 1:206
        Mag(count) = MAG(1,1,count);
        Phase(count) = PHASE(1,1,count);
    end

```

```

F = W/2/pi;
subplot(2,1,1)
hold off
plot(F,20*log10(Mag))
title('Frequency Response of the Harmonic Kalman Filter')
xlabel('Frequency (Hz)')
ylabel('Amplitude Ratio (dB)')
grid on
xlim([0 1000])
ylim([-40 10])
subplot(2,1,2)
hold off
plot(F,Phase)
xlabel('Frequency (Hz)')
ylabel('Phase (deg)')
grid on
xlim([0 1000])
ylim([-90 45])
set(gca,'ytick',-90:45:45)

```

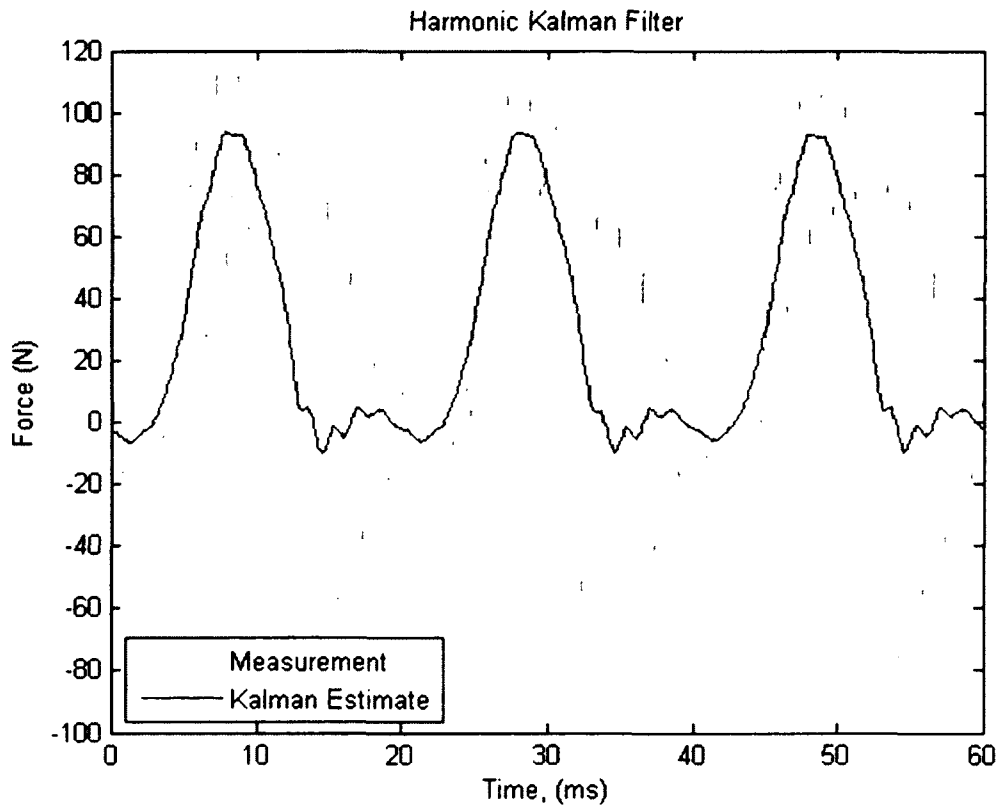


Figure E.21 -Implementation of the harmonic Kalman filter

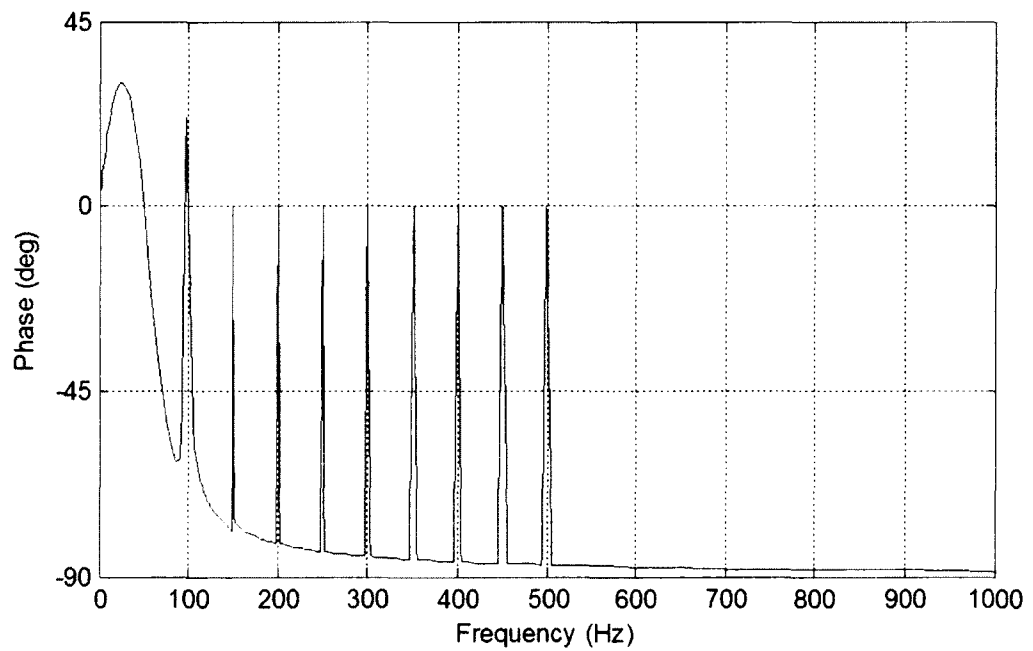
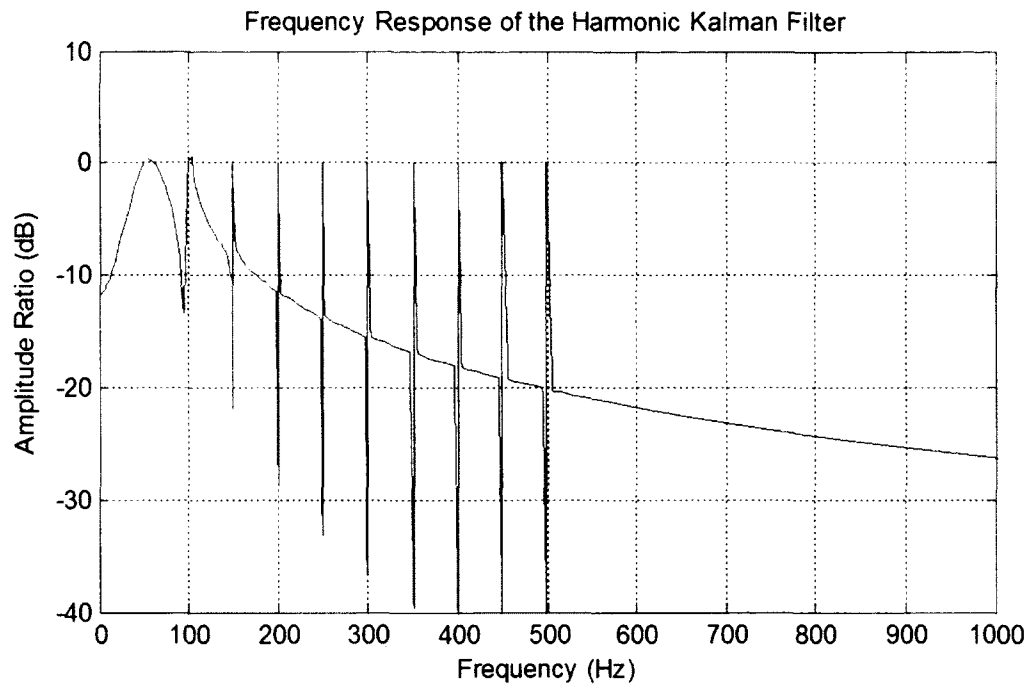


Figure E. 22 - Bode plot of the harmonic Kalman filter

Example Code for Implementing the Dynamic Chip Load Filter

Dynamic Chip Load Filter

```
clear
load hrad_3600_h2.mat % contains data and a cursor structure
clc
```

Transform data to force

```
% Parameters
N = length(data2);
fs = 10240;
t = (0:N-1)./fs;
force = (data2-501)./1.64; % lbf

% Plotting
figure(1)
hold off
plot(1000.*t-55,force,'color',[0.65 0.65 0.65])
title('Smart Tool - 3600 RPM, Half Immersion, Radial')
xlabel('Time (ms)');
ylabel('Measured "Force" (lbf)')
xlim([0 1000*max(t)])
grid on
ylim([-25 30])
xlim([0 90])
```

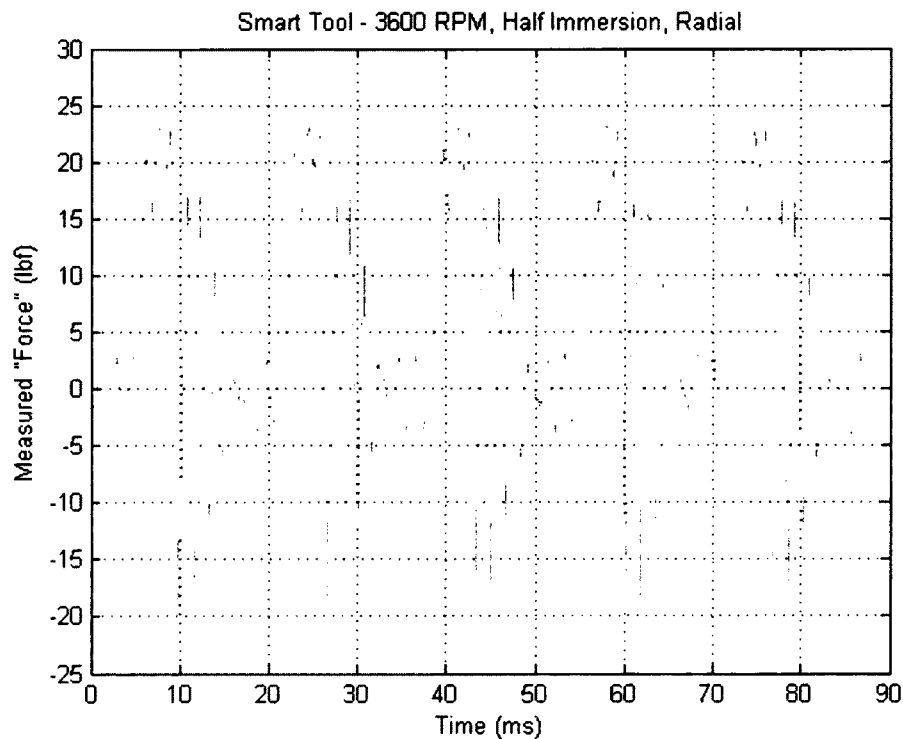


Figure E.23 – Example of Smart Tool data: Measured radial force, 3600 rpm

Time-Lagged Difference Sequence

```
% Interpolate onto better time axis
t2 = (0:2*N-1)./2./fs;
force2 = interp1(t,force,t2);

% Process Variables
process.AD = 0.125;
process.D = 0.75;
process.rpm = 3600;
process.f = 2;
process.immersion = 'half';
N = 171.5; % Samples per revolution

% Variation in Measurement (3600 rpm)
delta = zeros(1,length(force));
for i=1:length(force)
    if i > N
        delta(i) = force(i) - interp1(1:length(force),force,i-N);
    else
        delta(i) = 0;
    end
end
end
```

Static Chip Load Model

```
model = staticChip(force,process);
model = circshift(model,[10 0]);
```

Chipload Filter Force Estimate

```
Fest = delta + model(1:length(delta));
```

Chipload Filter Parameters - all on one plot

```
figure(3)
    hold off
    plot(1000.*t-55,force*4.448,'color',[0.65 0.65 0.65])
    hold on
    plot(1000.*t-55,delta*4.448,'r')
    plot(1000.*t-55,model(1:length(delta))*4.448,'b')
    xlim([0 38])
    ylim([-100 120])
    legend('Measured Force','Variation Sequence','Static Force
Model','location','southwest')
    title('Parameters for the Chipload Filter')
    xlabel('Time (ms)')
    ylabel('Force (N)')
```

```
figure(4)
    hold off
    plot(1000.*t-55,force*4.448,'color',[0.65 0.65 0.65])
```

```

hold on
plot(1000.*t-55, delta*4.448 + model(1:length(delta))*4.448,
'color',[0.7 0 0.7])
xlim([0 38])
ylim([-100 120])
legend('Measured Force','Model
Estimate','location','southwest')
title('Chipload Filter Estimate')
xlabel('Time (ms)')
ylabel('Force (N)')

```

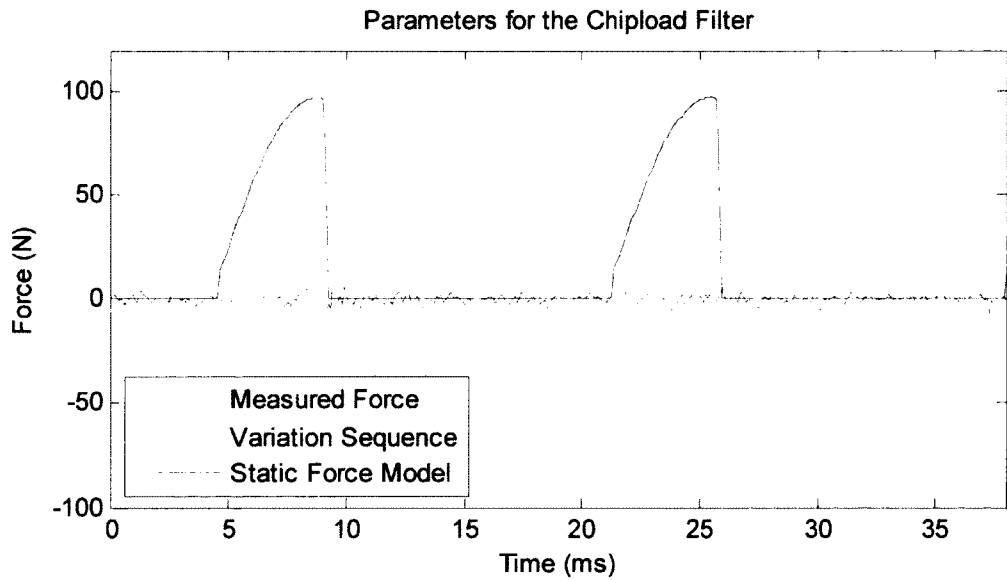
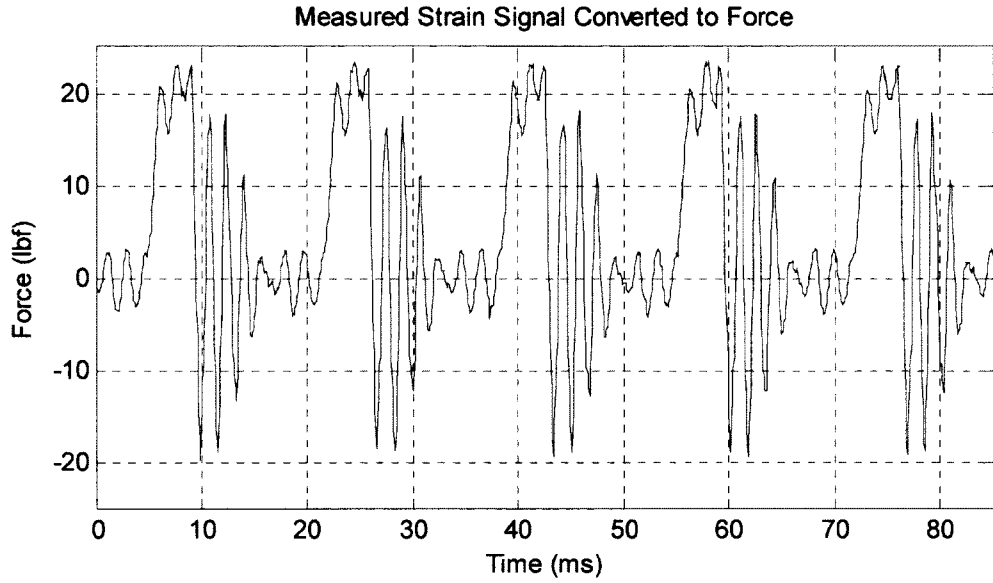


Figure E.24 – Parameters for the dynamic chip load filter

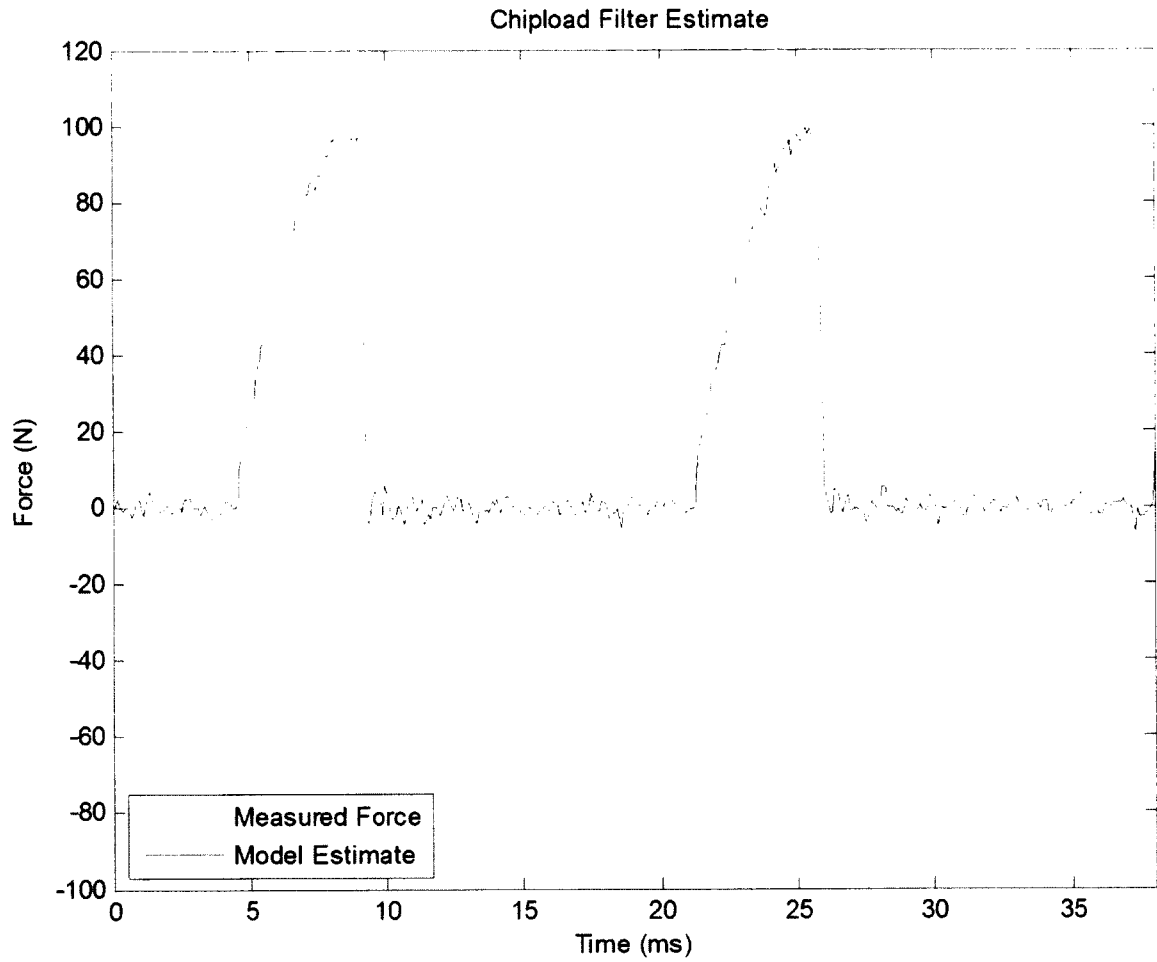


Figure E.25 – Implementation of the dynamic chip load filter

Example Code for Implementing Notch Filters

Design of IIR and FIR Notch Filters

```
clear
close all
load hrad_3600_h2.mat % contains data and a cursor structure
clc
```

Transform strain data to force

```
% Parameters
N = length(data2);
fs = 10240;
t = (0:N-1)./fs;
force = (data2-501)./1.64; % lbf

% Plotting
figure(1)
hold off
plot(1000.*t-55,force,'color',[0.65 0.65 0.65])
title('Smart Tool - 3600 RPM, Half Immersion, Radial')
xlabel('Time (ms)');
ylabel('Measured "Force" (lbf)')
xlim([0 1000*max(t)])
grid on
ylim([-25 30])
xlim([0 90])
```

Develop a Notch Filter

```
% Notch Filter Parameters
fn = 603;
w0 = 2*fn/fs; % Normalized Ringing Frequency (notch center)
bw = w0/3;
[b a] = iirnotch(w0,bw,-12);

% Filter Visualization
figure(2)
hold off
subplot(2,2,1)
[H F] = freqz(b,a,1024,fs);
plot(F,20*log10(abs(H)))
title('Notch Filter Magnitude Spectrum')
xlabel('Frequency (Hz)')
ylabel('Filter Gain (dB)')
xlim([0 5000])
grid on
subplot(2,2,3)
plot(F,atan2(imag(H),real(H)).*180./pi)
title('Notch Filter Phase Spectrum')
xlabel('Frequency (Hz)')
ylabel('Phase (deg)')
xlim([0 5000])
```

```

grid on
subplot(2,2,[2;4])
zplane(b,a)

% Implement filter
zf = filter(b,a,force);
zf2 = filtfilt(b,a,force);

```

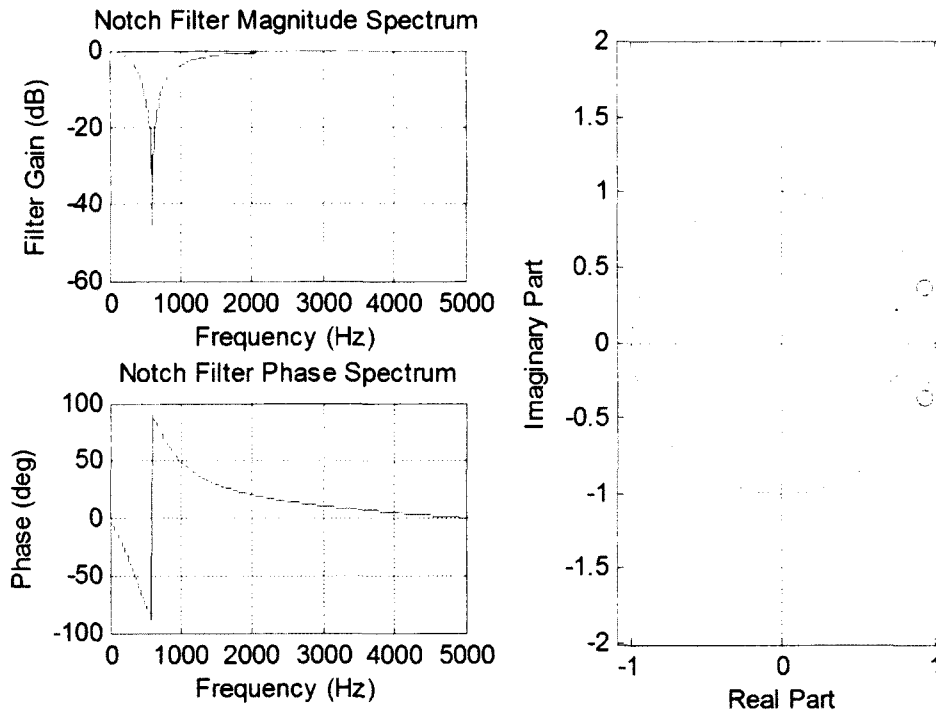


Figure E.26 – IIR notch filter z-plane and frequency response

Impulse Invariance FIR Notch Filter

```

% Notch Filter Impulse Response
x = zeros(100,1); x(1)=1;
h = filter(b,a,x);
Q = 30;

figure(3)
hold off
stem(0:Q-1,h(1:Q),'b*')
bFIR = h(1:Q);
title('Truncated Impulse Response')
xlabel('Data Index')
ylabel('Impulse Response')

figure(4)
zplane(roots(bFIR),[])
title('Zeros of the FIR Notch Filter')

figure(5)
[H, F] = freqz(bFIR,1,512,'half',10240);

```

```

subplot(2,1,1)
    plot(F,20*log10(abs(H)));
    title('Frequency Response of the FIR Notch Filter')
    xlabel('Frequency (Hz)')
    ylabel('Amplitude Ratio (dB)')
    xlim([0 5000])
    ylim([-60 0])
    grid on
subplot(2,1,2)
    plot(F,atan2(imag(H),real(H))*180/pi);
    xlabel('Frequency (Hz)')
    ylabel('Phase (deg)')
    xlim([0 5000])
    ylim([-180 180])
    set(gca,'ytick',-90:45:180)
    grid on

```

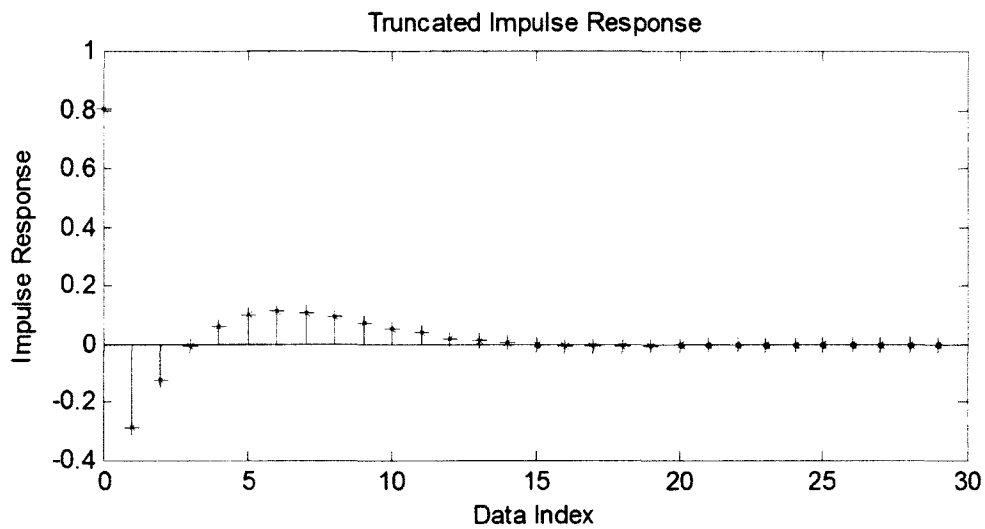


Figure E.27 – Truncated impulse response used to build the FIR notch filter

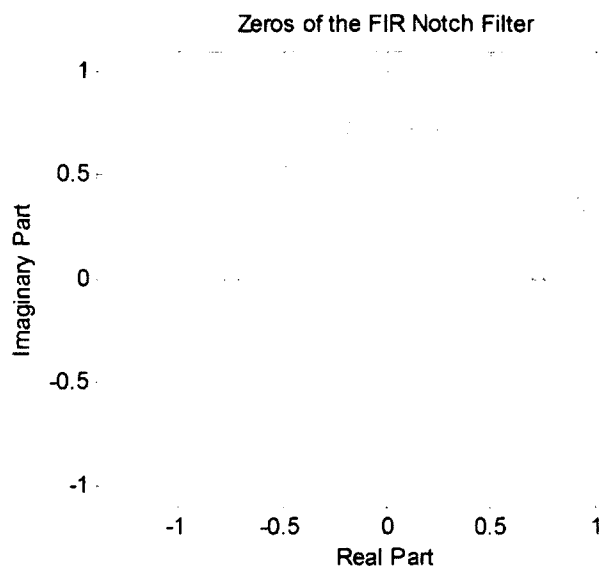


Figure E.28 - Zeros of the FIR notch filter

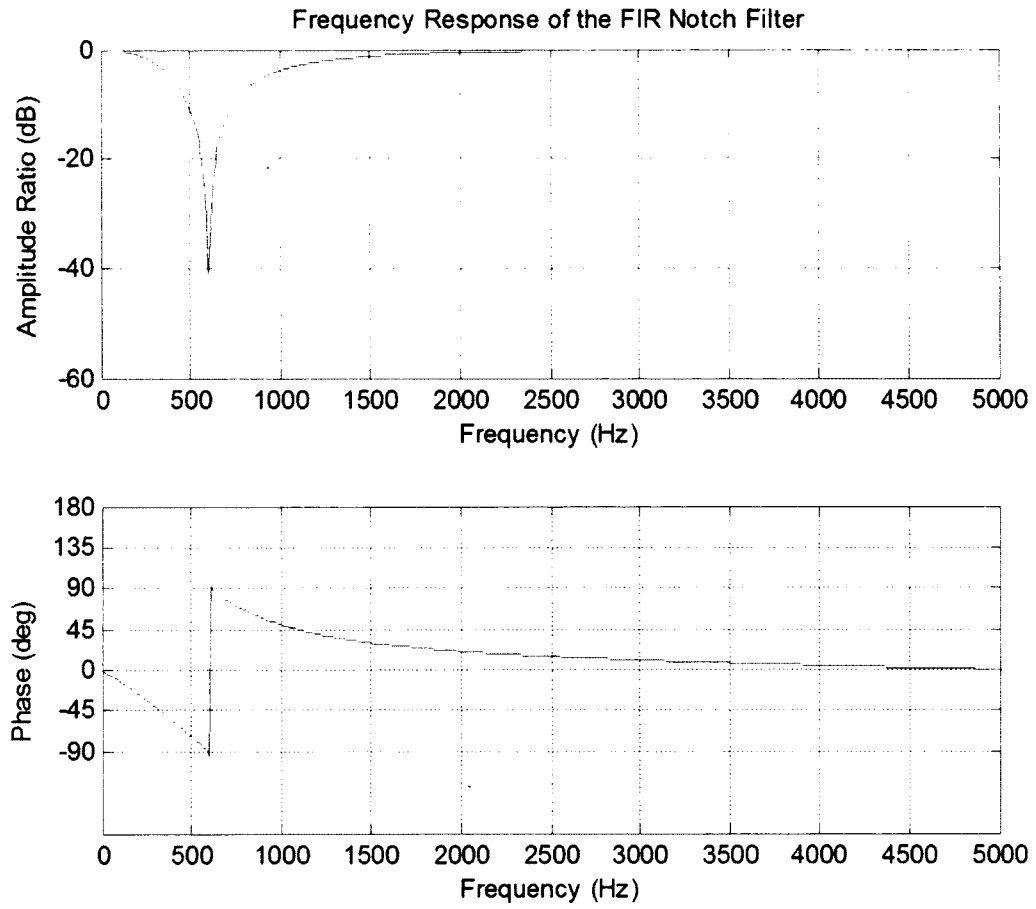


Figure E. 29 - Frequency response of the FIR notch filter

Notch Filter Implementation

```
figure(6)
hold off
plot(1000.*t-55,force*4.448,'color',[0.65 0.65 0.65])
hold on
plot(1000.*t-55,filter(b,a,force)*4.448,'color',[0.7 0
0.7],'linewidth',1)
title('IIR Notch Filter - 3600 RPM, Half Immersion, Radial')
xlabel('Time (ms)');
ylabel('Force (N)')
xlim([0 1000*max(t)])
ylim([-100 120])
xlim([0 35])
legend('Measurement','IIR Filter Estimate','location','southwest')
```

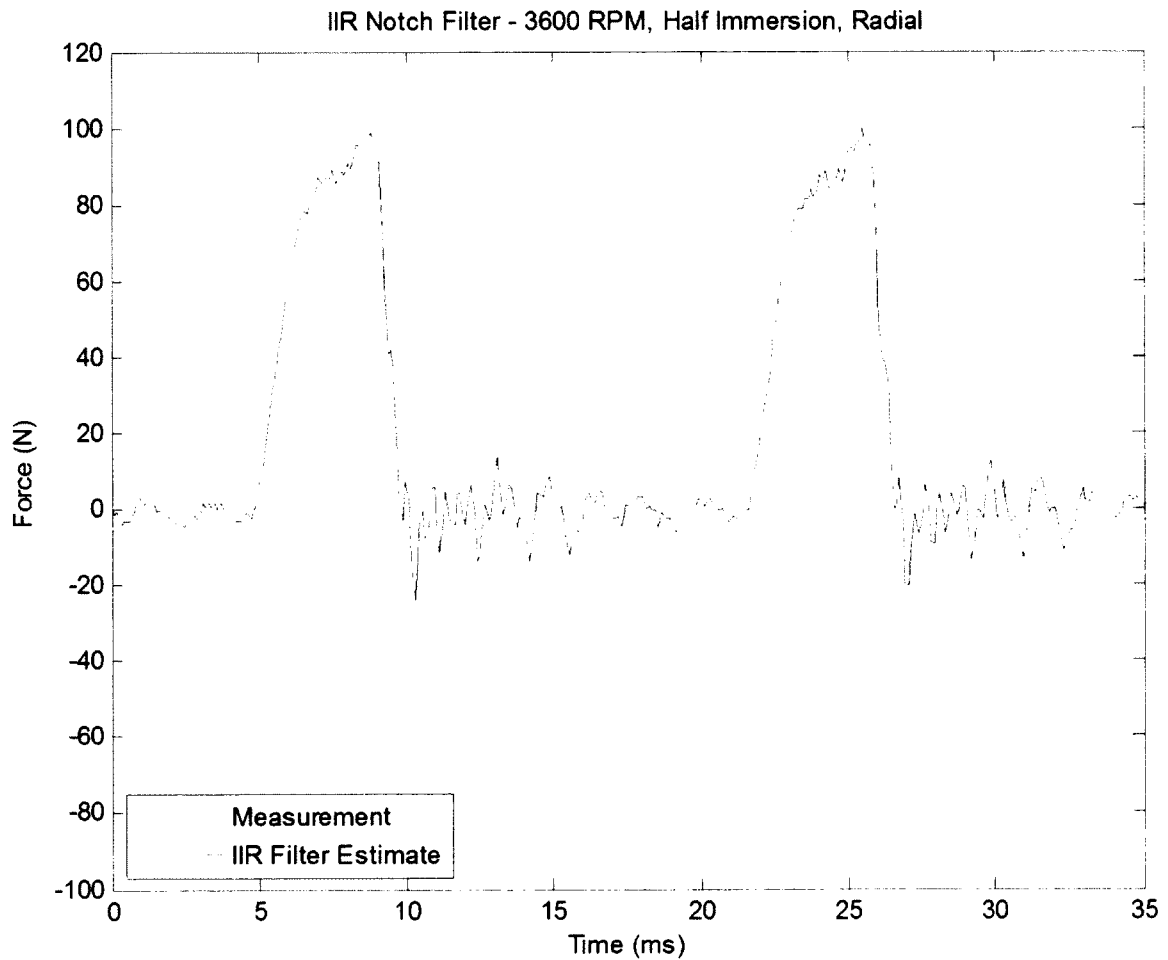


Figure E.30 – IIR notch filter implementation


```

figure(7)
hold off
plot(1000.*t-55,force*4.448,'color',[0.65 0.65 0.65])
hold on
plot(1000.*t-55,zf2*4.448,'color',[0.7 0 0.7],'linewidth',1)
title('IIR Notch Filter - Zero Phase Implementation')
xlabel('Time (ms)');
ylabel('Force (N)')
xlim([0 1000*max(t)])
ylim([-100 120])
xlim([0 35])
legend('Measurement','IIR Filter Estimate','location','southwest')

```

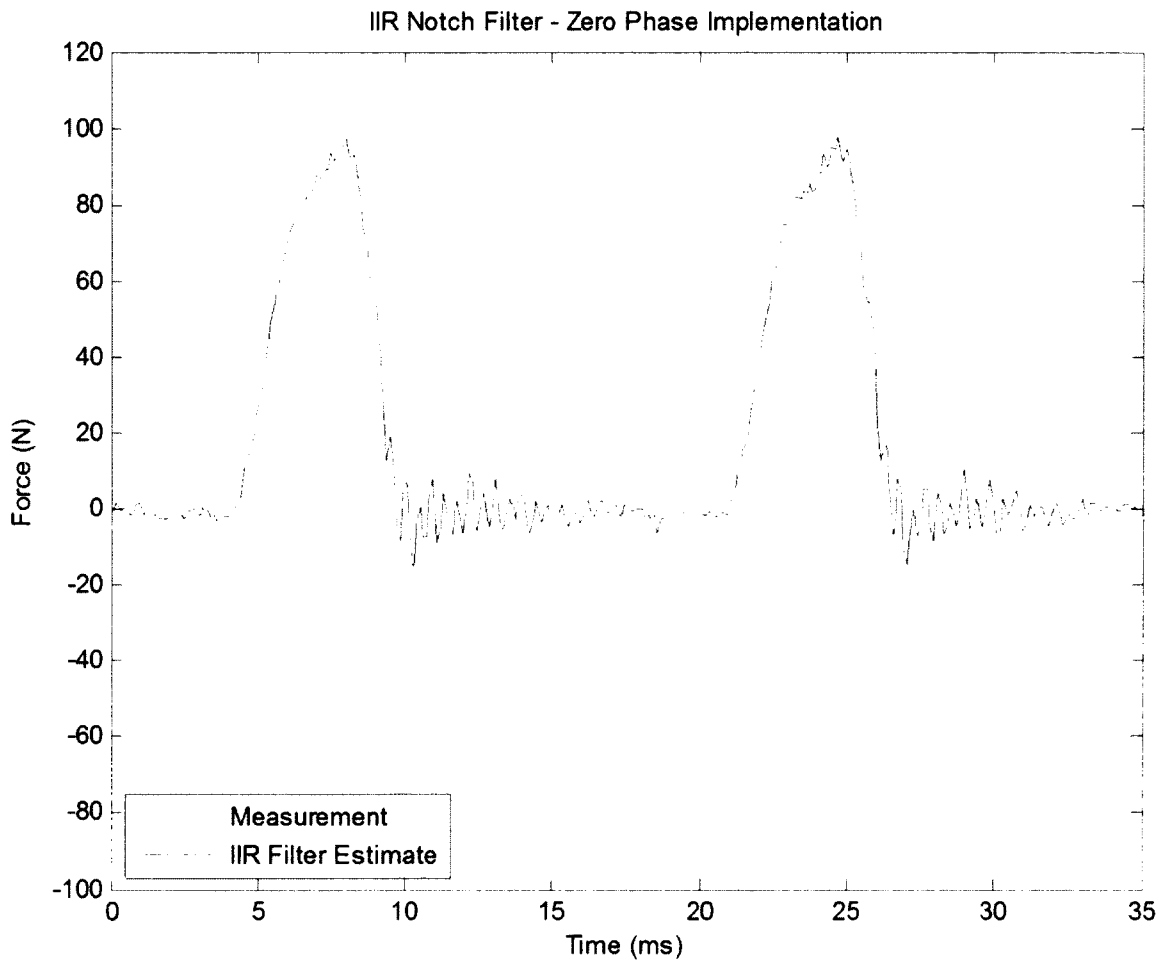


Figure E.31 - IIR notch filter, zero-phase implementation

Useful Subroutine for Power Spectral Estimation

This script uses Welch's Method to implement power spectral estimation. It allows the user to specify a data window, and to perform spectral smoothing either by band averaging, ensemble averaging, or a combination of the two.

```
function out = ssd(data,fs,varargin)
% Sample Spectral Density
%
% ssd(data,fs)
% ssd(data,fs,varargin)
% ssd(data,fs,'ens',n,'band',n,'method','method_name','window','window_name')
%
% out = ssd(data,fs,varargin)

out = struct('Sxx',[],'f',[],'CI',[],'BF',[],'bw',[]);

% Default Parameters
ens = 1;
band = 1;
method = 'fft';
window = 'boxcar';
plotting = 'half';

% Read in Additional Input Arguments
for k=1:length(varargin);

    switch lower(varargin{k})

        case{'ens'}
            if rem(varargin{k+1},1)~=0
                error('ensembles to average must be an integer');
            end
            ens = varargin{k+1};

        case{'band'}
            if rem(varargin{k+1},1)~=0
                error('data points to average per band must be an integer');
            end
            band = varargin{k+1};

        case{'method'}
            method = varargin{k+1};

        case{'window'}
            window = varargin{k+1};

        case{'whole'}
            plotting = 'whole';

    end

end

% Degrees of Freedom
DOF = 2*ens*band;

% Sampling Interval
delta = 1/fs;

% Make Sure that "data" is a Row Vector
[rows,cols] = size(data);
```

```

if rows > cols
    data = data';
end

% COMPUTE THE SAMPLE SPECTRAL DENSITY BY THE APPROPRIATE METHOD
switch(lower(method))

    case{'fft'} % Cooley-Tukey Method (by direct Fourier Transform)

        % Divide the Record Into Ensembles
        a = length(data); % Number of observations
        b = ens; % Number of ensembles to create
        l_ens = (a-rem(a,b))/ens; % Length of each ensemble row vector

        % Initialize the Sub-Record Vector
        sub_data = zeros(b,l_ens);

        % Initialize the sub_ssd array
        sub_ssd = zeros(b,floor(l_ens/2-2));

        % Break the Record into Sub-Records
        idx = 1;
        for i = 1:ens
            sub_data(i,:) = data(idx:idx+l_ens-1);
            idx = idx + l_ens;
        end

        % Select the right data window
        switch(lower(window))

            case{'boxcar'}
                N = l_ens;
                w = ones(1,N);

            case{'bartlett'}
                N = l_ens;
                n = 0:N-1;
                w = (2/(N-1)).*(((N-1)/2)-abs(n-((N-1)/2)));

            case{'hanning'}
                N = l_ens;
                w = hann(N)';

            case{'hamming'}
                N = l_ens;
                w = hamming(N)';

            case{'kaiser'}
                N = l_ens;
                w = kaiser(N)';

            case{'tukey'}
                N = l_ens;
                w = tukeywin(N)';

            case{'blackman'}
                N = l_ens;
                w = blackman(N)';

        end

        % COMPUTE THE SAMPLE SPECTRAL DENSITY FOR EACH SUB-RECORD
        L = N; % Length of the sub-record

```

```

    f = linspace(1/L, (fs/2)-(1/L), floor(L/2)-2); % Frequency vector
excluding 0, nyquist

    for i = 1:ens
        % Compute the fft
        g = sub_data(i,:); % Sub-record of the time series
        G = fft((g-mean(g)).*w,L)/N; % FFT normalized by no. observations
        G_hat = G(2:(floor(L/2)-1)); % FFT from 0 to the nyquist not
including endpoints
        A = real(G_hat);
        B = imag(G_hat);

        % Compute the sample spectral density of the sub-record
switch(lower(plotting))
    case{'half'}
        sub_ssd(i,:) = 2.*N.*delta.*(A.^2 + B.^2);
    case{'whole'}
        sub_ssd(i,:) = N.*delta.*(A.^2 + B.^2);
    end
    end

end

% COMPUTE THE ENSEMBLE-AVERAGE SAMPLE SPECTRAL DENSITY
ens_ssd = zeros(1,length(f)); % Initialize the vector
for j = 1:length(f)
    ens_ssd(j) = sum(sub_ssd(:,j))/ens; % Mean value of the sub-
estimates
end

% BAND-AVERAGE THE ENSEMBLE-AVERAGED ESTIMATE

    % Determine the number of bands
a = length(ens_ssd); % Length of Ensemble-Averaged SSD
b = band; % Number of points per band
n_bands = (a-rem(a,b))/band; % Number of bands

    % Define the band-averaged sample spectral density
band_ssd = zeros(1,n_bands); % Initialize vector
f2 = zeros(1,n_bands);
idx = 1; % Starting index
for n = 1:n_bands
    band_ssd(n) = sum(ens_ssd(idx:idx+band-1))/band;
    f2(n) = (f(idx)+f(idx+band-1))/2;
    idx = idx+band;
end

% Boost the Smoothed Estimate based on Windowing Error
num = var(data);
den = sum(band_ssd)*(f2(2)-f2(1));
BF = num/den;

% Return the Smoothed Estimate
s_bar = BF*band_ssd;

case{'prepost'} % Pre-whiten, Post-color

    % Compute the First Difference
y = diff1(data);

    % Divide the Record Into Ensembles
a = length(y); % Number of observations
b = ens; % Number of ensembles to create
l_ens = (a-rem(a,b))/ens; % Length of each ensemble row vector

```

```

% Initialize the Sub-Record Vectors
suby = zeros(b,l_ens);

% Break the Records into Sub-Records
idx = 1;
for i = 1:ens
    suby(i,:) = y(idx:idx+l_ens-1);
    idx = idx + l_ens;
end

% Compute the Fourier Transform of x(t) and y(t)
delta = 1/fs;
N = l_ens; % Number of observations per
sub-record
L = N; % Length of the sub-record
f = linspace(1/L, (fs/2)-(1/L), floor(L/2)-2); % Frequency vector
excluding 0, nyquist

% Initialize the auto-spectral density array
Syy = zeros(b, floor((L/2)-2));

for n = 1:ens

    % Compute the Fourier Coefficients
    yn = suby(n,:); % Sub-record of the time series y(t)

    Gy = fft(yn-mean(yn),L)/N; % FFT normalized by no. observations
    Gy_hat = Gy(2:floor(L/2)-1); % FFT from 0 to the nyquist not
including endpoints

    Ay = real(Gy_hat);
    By = imag(Gy_hat);

    % Compute the Auto-Spectral Density of the Filtered t.s.
    Syy(n,:) = 2.*N.*delta.*(Ay.^2 + By.^2);

end

% Compute the Ensemble-Averages
Syy_ens = zeros(1,length(f)); % Initialize the vector

for j = 1:length(f)
    Syy_ens(j) = sum(Syy(:,j))/ens; % Mean value of the sub-estimates
end

% Band-Average the Ensemble-Averaged Estimates

% Determine the number of bands
a = length(Syy_ens); % Length of Ensemble-Averaged SSD
b = band; % Number of points per band
n_bands = (a-rem(a,b))/band; % Number of bands

% Define the band-averaged Spectral Densities
Syy_band = zeros(1,n_bands); % Initialize the vector
f2 = zeros(1,n_bands); % Initialize frequency vector

idx = 1; % Starting index
for n = 1:n_bands
    Syy_band(n) = sum(Syy_ens(idx:idx+band-1))/band;
    f2(n) = (f(idx)+f(idx+band-1))/2;
    idx = idx+band;
end

```

```

        % Smooted Estimates of the 1-Sided Spectrum
        Syy = Syy_band;

    % Post-Color the White Spectrum
    num = Syy;
    den = 4.*sin(pi.*f2.*delta).^2;
    s_bar = num./den;
    BF = den;

end

% Confidence Interval
[x1 x2] = chi2('two',DOF,'percent',95);
CI = [DOF/x2, DOF/x1];

% Return Parameters
switch(lower(plotting))
    case{'half'}
        out.f = f2;                % Smoothed Frequency Vector
        out.bw = f2(2)-f2(1);      % Bandwidth
        out.Sxx = s_bar;          % Sample Spectral Density
        out.CI = CI;              % Confidence Interval
        out.BF = BF;              % Boost Factor/ Filter Spectrum
    case{'whole'}
        delf = f2(2)-f2(1);
        N = 2*length(s_bar)+1;
        fs = delf*N;
        m = 0:N;
        f = m./N.*fs;
        out.f = f;                % Smoothed Frequency Vector
        out.bw = f2(2)-f2(1);      % Bandwidth
        out.Sxx = [NaN s_bar NaN fliplr(s_bar)]; % Sample Spectral Density
        out.CI = CI;              % Confidence Interval
        out.BF = BF;              % Boost Factor/ Filter Spectrum
end

```