VALIDATION OF A DC-DC BOOST CIRCUIT MODEL AND CONTROL ALGORITHM

Dissertation

Submitted to

The School of Engineering of the

UNIVERSITY OF DAYTON

In Partial Fulfillment of the Requirements for

The Degree of

Doctor of Philosophy in Electrical Engineering

By

Jon Tomas Zumberge

Dayton, Ohio

August, 2015
ABSTRACT

VALIDATION OF A DC-DC BOOST CIRCUIT MODEL AND CONTROL ALGORITHM

Name: Zumberge, Jon Tomas
University of Dayton

Advisor: Dr. Raúl Ordóñez

Cost and performance requirements are driving military and commercial systems to highly integrated, optimized systems which require more sophisticated, highly complex controls. To realize benefits and make confident decisions, the validation of both plant and control models becomes critical. To quickly develop controls for these systems, it is beneficial to develop models and determine the uncertainty of those models to predict performance and stability. A process of model and control algorithm validation for a dc-dc boost converter circuit based on acceptance sampling is presented here. The verification and validation process described in this dissertation is based on MIL-STD 3022 with emphasis on requirements settings and the validation process. To minimize the cost of experimentation and simulation, design of experiments is used extensively to limit the amount of data taken without losing information.

The key contribution of this dissertation include the process for model and control algorithm validation specifically a method for decomposing the problem into a model validation stage and a control algorithm validation stage. The other contributions include a metric for differentiating between strong validation data and weak validation data, projection of model and control uncertainty
limits to areas where experimental data has not been taken, and an improved means of tolerance interval calculation for non-parametric distributions.
In memory of Rita Zumberge.
ACKNOWLEDGMENTS

I would like to thank my advisor Dr. Raúl Ordóñez for his guidance and encouragement throughout these studies. I would like to thank my committee members Dr. Malcolm Daniels, Dr. Keigo Hirakawa, and Dr. John Doty for their suggestions and guidance throughout these studies. I would like to thank the Air Force for funding this work. I would like to thank Mr. John Mersch for his assistance in automating the data collection and creating the PCB board for the dc-dc circuit. Finally, I thank my wife for her support.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABSTRACT</td>
<td>iii</td>
</tr>
<tr>
<td>DEDICATION</td>
<td>v</td>
</tr>
<tr>
<td>ACKNOWLEDGMENTS</td>
<td>vi</td>
</tr>
<tr>
<td>LIST OF FIGURES</td>
<td>xi</td>
</tr>
<tr>
<td>LIST OF TABLES</td>
<td>xiv</td>
</tr>
<tr>
<td>I. INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>II. BACKGROUND</td>
<td>4</td>
</tr>
<tr>
<td>2.1 Model Validation</td>
<td>4</td>
</tr>
<tr>
<td>2.1.1 Feature Examination</td>
<td>5</td>
</tr>
<tr>
<td>2.1.2 P-Box Type Techniques</td>
<td>6</td>
</tr>
<tr>
<td>2.1.3 Mean Value Techniques</td>
<td>7</td>
</tr>
<tr>
<td>2.1.4 Bayesian Hypothesis Testing (interval based)</td>
<td>7</td>
</tr>
<tr>
<td>2.1.5 Whiteness Tests</td>
<td>8</td>
</tr>
<tr>
<td>2.1.6 Anderson-Darling Tests</td>
<td>9</td>
</tr>
<tr>
<td>2.2 System Identification</td>
<td>10</td>
</tr>
<tr>
<td>2.3 Robust Controls</td>
<td>11</td>
</tr>
<tr>
<td>2.4 Statistics</td>
<td>12</td>
</tr>
<tr>
<td>2.4.1 Risk</td>
<td>12</td>
</tr>
<tr>
<td>2.4.2 Experimental Designs</td>
<td>13</td>
</tr>
<tr>
<td>2.4.3 Sampling</td>
<td>14</td>
</tr>
<tr>
<td>2.4.4 Regression Analysis</td>
<td>16</td>
</tr>
<tr>
<td>2.4.5 Statistical Intervals</td>
<td>17</td>
</tr>
<tr>
<td>2.5 Dynamic Measurement Uncertainty Quantification and Propagation</td>
<td>19</td>
</tr>
<tr>
<td>2.6 DC-DC Boost Converter Model</td>
<td>20</td>
</tr>
</tbody>
</table>
III. PROBLEM DEFINITION ......................................................... 25

IV. GENERALIZED SOLUTION .................................................. 28

4.1 Model and Control Uncertainty Analysis ................................ 28

4.1.1 Uncertainty Development ............................................... 28

4.1.2 Lipschitz Constant Estimation and Distance Evaluations ....... 35

4.1.3 Uncertainty Limits ..................................................... 40

4.1.4 Total Uncertainty Estimates ......................................... 41

4.2 Simulation Setup ......................................................... 42

V. DC-DC CIRCUIT MODEL APPLICATION .................................. 44

5.1 DC-DC Boost Circuit Model and Control Development ............ 44

5.1.1 Model Development ................................................... 44

5.2 Simulation Setup ......................................................... 46

5.3 Control Development ................................................... 47

5.3.1 Backstepping Control ................................................ 48

5.3.2 Beta Modification ..................................................... 52

5.3.3 Augmented Backstepping Control .................................. 52

5.4 Domains and Data Sets .................................................. 53

5.5 Process ................................................................. 57

5.5.1 Problem Statement ................................................... 58

5.5.2 Requirements ......................................................... 59

5.5.3 Model Verification and Validation .................................. 68

5.5.4 Verification and Validation Issues and Recommendations .... 75

5.6 Final Prediction and Test ................................................ 75

VI. EXAMPLE PROBLEMS ....................................................... 77

6.1 Ideal Experiment, Ideal Plant, Ideal Control (Example Problem 1) 77

6.2 Diode Loss Experiment, Ideal Plant, Ideal Control (Example Problem 2) 80

6.3 Diode Loss Experiment, Diode Loss Plant, Ideal Control (Example Problem 3) 87

6.4 Diode Loss Experiment (with parameter changes), Diode Loss Plant, Diode Loss Control (Example Problem 4) 93

6.5 Full Plant Experiment (No Time Delay), Full Plant, Full Control (Example Problem 5) 95

6.6 Simple Experiment (with filtered data), Simple Plant, Simple Control (Example Problem 6) 98

VII. HARDWARE DESIGN AND UNCERTAINTY ANALYSIS .............. 101
## LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Physical Model of DC-DC Boost Converter</td>
<td>21</td>
</tr>
<tr>
<td>4.1</td>
<td>Example of using the Lipschitz prediction</td>
<td>34</td>
</tr>
<tr>
<td>4.2</td>
<td>Example of weak validation data (red x indicates farthest point), distance = 0.0775</td>
<td>38</td>
</tr>
<tr>
<td>4.3</td>
<td>Example of stronger validation data (red x indicates farthest point), distance = 0.5240</td>
<td>39</td>
</tr>
<tr>
<td>4.4</td>
<td>Integrator in the loop simulation model (closed loop model)</td>
<td>42</td>
</tr>
<tr>
<td>4.5</td>
<td>Integrator out of the loop simulation model (open loop model)</td>
<td>43</td>
</tr>
<tr>
<td>5.1</td>
<td>Average value model of dc-dc boost circuit</td>
<td>45</td>
</tr>
<tr>
<td>5.2</td>
<td>Details of the dc-dc boost circuit open loop model</td>
<td>47</td>
</tr>
<tr>
<td>5.3</td>
<td>Details of the dc-dc boost circuit closed loop model</td>
<td>48</td>
</tr>
<tr>
<td>5.4</td>
<td>Invariant set (green) with beta modification of $\beta_c = 1$</td>
<td>53</td>
</tr>
<tr>
<td>5.5</td>
<td>Invariant set (green) with beta modification of $\beta_c = 15$</td>
<td>54</td>
</tr>
<tr>
<td>5.6</td>
<td>Plot of the output voltage ($V_o$) of a typical test vector in $D_t$</td>
<td>55</td>
</tr>
<tr>
<td>5.7</td>
<td>Plot of the inductor current ($I_l$) of a typical test vector in $D_t$. The current command is the same as $\alpha_c$.</td>
<td>56</td>
</tr>
<tr>
<td>5.8</td>
<td>Plot of the load current ($I_o$) of a typical test vector in $D_t$.</td>
<td>57</td>
</tr>
<tr>
<td>5.9</td>
<td>Plot of the source voltage ($V_i$) of a typical test vector in $D_t$. Commanded input voltage was 3 V.</td>
<td>58</td>
</tr>
</tbody>
</table>
5.10 Plot of the output voltage ($V_o$) of a typical test vector in $D_v$. . . . . . . . . . . 59

5.11 Plot of the inductor current ($I_l$) of a typical test vector in $D_v$. The current command is the same as $\alpha_c$. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 60

5.12 Plot of the load current ($I_o$) of a typical test vector in $D_v$. . . . . . . . . . . . . 61

5.13 Plot of the source voltage ($V_i$) of a typical test vector in $D_v$. Commanded input voltage was 3.0694 V. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 62

5.14 MIL-STD 3022 Documentation Outline. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . ...
<table>
<thead>
<tr>
<th>Chapter Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>9.2</td>
<td>Original data, invariant set</td>
<td>138</td>
</tr>
<tr>
<td>9.3</td>
<td>Invariant set of the Fmincon Parameter Set</td>
<td>139</td>
</tr>
<tr>
<td>9.4</td>
<td>Invariant set of the GA parameter set</td>
<td>140</td>
</tr>
<tr>
<td>9.5</td>
<td>Invariant set of the Derivative Output parameter set</td>
<td>141</td>
</tr>
<tr>
<td>9.6</td>
<td>Simulated invariant set of the Fmincon parameter set</td>
<td>142</td>
</tr>
<tr>
<td>9.7</td>
<td>Simulated invariant set for Example Problem 5</td>
<td>143</td>
</tr>
<tr>
<td>B.1</td>
<td>Non-Parametric Tolerance Interval (One-Sided) for the Max Points Method ($\alpha = 0.8$)</td>
<td>159</td>
</tr>
<tr>
<td>B.2</td>
<td>Non-Parametric Tolerance Interval (One-Sided) for the Baseline Method ($\alpha = 0.8$)</td>
<td>160</td>
</tr>
<tr>
<td>B.3</td>
<td>Non-Parametric Tolerance Interval (One-Sided) for the Downsampling Method ($\alpha = 0.8$)</td>
<td>160</td>
</tr>
<tr>
<td>B.4</td>
<td>Non-Parametric Tolerance Interval (One-Sided) for Max Points Method ($\alpha = 0.95$)</td>
<td>161</td>
</tr>
<tr>
<td>B.5</td>
<td>Non-Parametric Tolerance Interval (One-Sided) for the Baseline Method ($\alpha = 0.95$)</td>
<td>161</td>
</tr>
<tr>
<td>B.6</td>
<td>Non-Parametric Tolerance Interval (One-Sided) for the Downsampling Method ($\alpha = 0.95$)</td>
<td>162</td>
</tr>
<tr>
<td>Table Number</td>
<td>Description</td>
<td>Page</td>
</tr>
<tr>
<td>-------------</td>
<td>------------------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>6.1</td>
<td>Model Uncertainty Tolerance Intervals of Data Set $D_t$ for Example Problem 1</td>
<td>79</td>
</tr>
<tr>
<td>6.2</td>
<td>Model Uncertainty Tolerance Intervals of Data Set $D_v$ for Example Problem 1</td>
<td>79</td>
</tr>
<tr>
<td>6.3</td>
<td>Control Uncertainty Tolerance Intervals of Data Set $D_t$ for Example Problem 1</td>
<td>79</td>
</tr>
<tr>
<td>6.4</td>
<td>Control Uncertainty Tolerance Intervals of Data Set $D_v$ for Example Problem 1</td>
<td>80</td>
</tr>
<tr>
<td>6.5</td>
<td>Model Uncertainty Functional Estimate, Measured, and Analytical Uncertainty Comparisons for Data Set $D_t$ for Example Problem 2</td>
<td>81</td>
</tr>
<tr>
<td>6.6</td>
<td>$\delta_{m1}$, Model Uncertainty, Tuning Results for Data Set $D_t$ for Example Problem 2</td>
<td>82</td>
</tr>
<tr>
<td>6.7</td>
<td>Regression Diagnostics for $\delta_{m1}$ for Data Set $D_t$ for Example Problem 2</td>
<td>82</td>
</tr>
<tr>
<td>6.8</td>
<td>$\delta_{m2}$, Model Uncertainty, Results for Data Set $D_t$ for Example Problem 2</td>
<td>82</td>
</tr>
<tr>
<td>6.9</td>
<td>Regression Diagnostics for $\delta_{m2}$ for Data Set $D_t$ for Example Problem 2</td>
<td>82</td>
</tr>
<tr>
<td>6.10</td>
<td>Model Uncertainty Tolerance Intervals for Data Set $D_t$ for Example Problem 2</td>
<td>82</td>
</tr>
<tr>
<td>6.11</td>
<td>Model Hypothesis Testing for Example Problem 2</td>
<td>83</td>
</tr>
<tr>
<td>6.12</td>
<td>Uncertainty Function Estimate, Lipschitz Estimate, Measured Value, and Analytical Uncertainty Comparisons for Data Set $D_v$ for Example Problem 2</td>
<td>84</td>
</tr>
<tr>
<td>6.13</td>
<td>Euclidean Distance Between Data Sets Using $</td>
<td></td>
</tr>
<tr>
<td>6.14</td>
<td>Model Uncertainty Tolerance Intervals for Integrator in the Loop Model with Data Set $D_t$ for Example Problem 2</td>
<td>85</td>
</tr>
<tr>
<td>Section</td>
<td>Title</td>
<td>Page</td>
</tr>
<tr>
<td>---------</td>
<td>----------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>6.15</td>
<td>Control Uncertainty Tolerance Intervals for Data Set $D_t$ for Example Problem 3</td>
<td>87</td>
</tr>
<tr>
<td>6.16</td>
<td>$\delta_{c1}$ Tuning Results for Data Set $D_t$ for Example Problem 3</td>
<td>88</td>
</tr>
<tr>
<td>6.17</td>
<td>Regression Diagnostic Table for $\delta_{c1}$ for Data Set $D_t$ for Example Problem 3</td>
<td>88</td>
</tr>
<tr>
<td>6.18</td>
<td>$\delta_{c2}$ Tuning Results for Data Set $D_t$ for Example Problem 3</td>
<td>89</td>
</tr>
<tr>
<td>6.19</td>
<td>Regression Diagnostic Table for $\delta_{c2}$ for Data Set $D_t$ for Example Problem 3</td>
<td>89</td>
</tr>
<tr>
<td>6.20</td>
<td>Control Uncertainty Tolerance Intervals for Data Set $D_v$ for Example Problem 3</td>
<td>89</td>
</tr>
<tr>
<td>6.21</td>
<td>$\delta_{c1}$ Polynomial for Data Set $D_v$ for Example Problem 3</td>
<td>90</td>
</tr>
<tr>
<td>6.22</td>
<td>Regression Diagnostic Table for $\delta_{c1}$ from Data Set $D_v$ for Example Problem 3</td>
<td>90</td>
</tr>
<tr>
<td>6.23</td>
<td>$\delta_{c2}$ Polynomial for Data Set $D_v$ for Example Problem 3</td>
<td>90</td>
</tr>
<tr>
<td>6.24</td>
<td>Regression Diagnostic Table for $\delta_{c2}$ from Data Set $D_v$ for Example Problem 3</td>
<td>91</td>
</tr>
<tr>
<td>6.25</td>
<td>Control Uncertainty Function Estimate, Lipschitz Estimate, and Measurement Uncertainty Comparison for Data Set $D_s$ for Example Problem 3</td>
<td>91</td>
</tr>
<tr>
<td>6.26</td>
<td>Hypothesis Testing Results of the Control Uncertainty Limits for Data Set $D_s$ for Example Problem 3</td>
<td>92</td>
</tr>
<tr>
<td>6.27</td>
<td>Control Uncertainty Tolerance Intervals for Data Set $D_s$ for Example Problem 3</td>
<td>92</td>
</tr>
<tr>
<td>6.28</td>
<td>Euclidean distance between data sets (Control Distances) $</td>
<td></td>
</tr>
<tr>
<td>6.29</td>
<td>Parameter Optimization for Fourth Example Problem</td>
<td>93</td>
</tr>
<tr>
<td>6.30</td>
<td>Model Uncertainty Tolerance Intervals of Data Set $D_t$ for Example Problem 4</td>
<td>94</td>
</tr>
<tr>
<td>6.31</td>
<td>Regression Diagnostic Table for $\delta_{m1}$ from Data Set $D_t$ for Example Problem 4</td>
<td>94</td>
</tr>
<tr>
<td>6.32</td>
<td>Regression Diagnostic Table for $\delta_{m2}$ from Data Set $D_t$ for Example Problem 4</td>
<td>94</td>
</tr>
</tbody>
</table>
6.33 Uncertainty Function Estimate, Lipschitz Estimate, and Measured Comparison using Data Set $D_v$ for Example Problem 4. .................................................. 95

6.34 Hypothesis Test Results of Model Uncertainty for Data Set $D_v$ for Example Problem 4. .................................................................................................................................................................................. 95

6.35 Parameter Optimization for Example Problem 5, Difference between the Experimental Parameter Values and the Optimized Parameter Values. ................................................................. 96

6.36 Model Uncertainty Tolerance Intervals for Data Set $D_t$ for Example Problem 5. .................................................................................................................................................................................. 97

6.37 Model Uncertainty Tolerance Intervals for Data Set $D_v$ for Example Problem 5. .................................................................................................................................................................................. 97

6.38 Hypothesis Testing Results of Model Uncertainty for Data Set $D_v$ for Example Problem 5. .................................................................................................................................................................................. 97

6.39 Control Uncertainty Tolerance Intervals for Data Set $D_a$ for Example Problem 5. .................................................................................................................................................................................. 98

6.40 Model Uncertainty Tolerance Intervals for Data Set $D_t$ for Example Problem 6. .................................................................................................................................................................................. 100

6.41 Model Uncertainty Tolerance Intervals for Data Set $D_v$ for Example Problem 6. .................................................................................................................................................................................. 100

7.1 Capacitor B24 capacitance and series resistance as a function of frequency. .................................................................................................................................................................................. 106

7.2 Inductor B15 inductance and series resistance as a function of frequency. .................................................................................................................................................................................. 106

7.3 Uncertainty Analysis of Components for DC-DC Boost Circuit. .................................................................................................................................................................................. 110

7.4 Range of Values used for Optimization Studies of Parameters. .................................................................................................................................................................................. 110

7.5 Values used in the current uncertainty calculation. .................................................................................................................................................................................. 112

8.1 Parameter Sets used In the Model Validation Comparison. .................................................................................................................................................................................. 116

8.2 Comparison of Different Optimization Routines. .................................................................................................................................................................................. 116

8.3 $\delta_{m1}$ Polynomial for Data Set $D_t$ for Model Fmincon w/ $J_x$ Cost Function. .................................................................................................................................................................................. 117

8.4 Regression Diagnostic Table for $\delta_{m1}$ for Data Set $D_t$ for Model Fmincon w/ $J_x$ Cost Function. .................................................................................................................................................................................. 118
8.23 Regression Diagnostic Table for $\delta_{m2}$ for Data Set $D_v$ for Model GA.

8.24 Distance Calculation used for Prediction of Data Set $D_f$.

8.25 Hypothesis Testing Results of Model Uncertainty for Data Set $D_f$.

9.1 Control Uncertainty Limit Calculations for Data Set $D_t$.

9.2 $\delta_{c1}$, Augmented Controller for Data Set $D_t$ for Model fmincon w/ $J_x$ cost function.

9.3 $\delta_{c1}$, Augmented Controller for Data Set $D_t$ for Model GA.

9.4 Regression Diagnostic Table for $\delta_{c1}$ (Augmented Controller) for Model fmincon w/ $J_x$ cost function.

9.5 Regression Diagnostic Table for $\delta_{c1}$ (Augmented Controller) for Model GA.

9.6 $\delta_{c1}$ Polynomial for Data Set $D_u$ for Model fmincon w/ $J_x$ cost function.

9.7 Regression Diagnostic Table for $\delta_{c1}$ for Data Set $D_u$ for Model fmincon w/ $J_x$ cost function.

9.8 $\delta_{c1}$ Polynomial for Data Set $D_u$ for Model GA.

9.9 Regression Diagnostic Table for $\delta_{c1}$ for Model GA.

9.10 Distance Calculation for Prediction of Data Set $D_a$.

9.11 Hypothesis Testing Results of Control Uncertainty for Data Set $D_a$.

9.12 $\delta_{c1}$ Polynomial for Data Set $D_a$ for Model fmincon w/ $J_x$ Cost Function.

9.13 $\delta_{c1}$ Polynomial for Data Set $D_a$ for Model GA.

9.14 Regression Diagnostic Table for $\Delta_{c1}$ for Data Set $D_a$ for Model fmincon w/ $J_x$ Cost Function.

9.15 Regression Diagnostic Table for $\Delta_{c1}$ for Data Set $D_a$ for Model GA.

9.16 Distance Calculation for Prediction of $D_f$. 
NOMENCLATURE

α risk allowed for Type I error
α_c current command
α_f inverse of the filter constant
β risk allowed for Type II error
β_c beta modification coefficient
Δ uncertainty limit
δ uncertainty signal
Δ_{ci} control uncertainty limit
δ_{ci} control uncertainty signal of $i^{th}$ differential equation
Δ_{mi} model uncertainty limit
δ_{mi} model uncertainty signal of $i^{th}$ differential equation
λ filter constant
λ_p mixing parameter for contaminated normal distribution
µ mean
ν degrees of freedom
Φ_{ji} resultant test data for data set j, differential equation i
σ population standard deviation
C capacitance
D_{sv} simulated validation data set
D_F second order term for diode drop
D_f future data set
D_s simulated future data set
d_T duty cycle
d_{Tg} duty cycle gain modifier
d_{To} duty cycle offset modifier
D_t tuning data set
D_v validation data set
e_1 error in output voltage
e_2 error in inductor current
F(x) cumulative distribution function
F_β cumulative beta distribution
f_i unforced dynamic of $i^{th}$ differential equation
G domain of all data sets
g_i forced dynamic of $i^{th}$ differential equation
I_l inductor current
I_o output current
\( J_\dot{x} \) cost function using derivative of error signals
\( J_x \) cost function using error signals
\( k_1 \) gain on voltage error
\( k_2 \) gain on inductor current error
\( L \) inductance
\( L_{ci} \) control uncertainty Lipschitz constant
\( L_{mi} \) model uncertainty Lipschitz constant
\( p \) derivative operator
\( P \) parameters
\( P() \) probability function
\( p_1 \) acceptable quality level of the model
\( p_2 \) rejectable quality level of the model
\( r \) equivalent resistive losses of switching components
\( R^2 \) coefficient of determination
\( R^2_{adj} \) adjusted coefficient of determination
\( R^2_{pred} \) prediction coefficient of determination
\( r_{ds} \) on resistance of n-MOSFET
\( r_{sns1} \) sense resistor one (used for measuring inductor current)
\( r_{sns2} \) sense resistor two (used for measuring output current)
\( r_c \) equivalent series resistance of capacitor
\( R_F \) on-resistance of diode
\( r_l \) equivalent series resistance of inductor
\( R_L \) load resistance
\( s \) disturbances
\( T_D \) control delay
\( u \) input
\( V \) Lyapunov candidate function
\( V_{oref} \) reference output voltage command
\( V_c \) voltage across capacitance of capacitor
\( V_F \) diode feedforward voltage drop
\( V_i \) input voltage
\( X \) vector of states
\( x_{ci} \) control differential
\( x_{mi} \) model differential
\( x_i \) experimental differential
<table>
<thead>
<tr>
<th>ACRONYMS</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>AFRL</td>
<td>Air Force Research Laboratory</td>
</tr>
<tr>
<td>AQL</td>
<td>acceptable quality level</td>
</tr>
<tr>
<td>ASME</td>
<td>American Society of Mechanical Engineers</td>
</tr>
<tr>
<td>CDF</td>
<td>cumulative distribution function</td>
</tr>
<tr>
<td>ESR</td>
<td>equivalent series resistance</td>
</tr>
<tr>
<td>GA</td>
<td>genetic algorithm</td>
</tr>
<tr>
<td>INVENT</td>
<td>Integrated Vehicle Energy Technology</td>
</tr>
<tr>
<td>LPF</td>
<td>low pass filter</td>
</tr>
<tr>
<td>PRESS</td>
<td>prediction error sum of squares</td>
</tr>
<tr>
<td>PWM</td>
<td>pulse width modulated</td>
</tr>
<tr>
<td>RQL</td>
<td>rejectable quality level</td>
</tr>
<tr>
<td>SRQ</td>
<td>system response quantity</td>
</tr>
</tbody>
</table>
CHAPTER I

INTRODUCTION

With the increase in complexity of systems fielded by both military and commercial entities, there is an ever growing need to “prove” these systems will perform to specifications in a stable manner. To address these issues, there is on-going work to include model based engineering earlier in the development cycle. One such program is the Air Force Research Laboratories (AFRL) Integrated Vehicle Energy Technology (INVENT) program [1], [2], and [3]. To realize the benefits of model based engineering, there is a need for validation of models, control algorithms, and software used in these complex systems at a reasonable cost [4], [5].

The goal of this work is to describe a novel methodology for nonlinear control algorithm and model validation. Here, the definition of control algorithm validation is to prove that with a prespecified amount of risk, a control algorithm will behave in a stable manner and meet performance specifications through use of a validated model and/or through experimental testing. The key contributions discussed in this work are first the model and control algorithm validation process (see Section 5.5), a means to calculate the differential equation uncertainty as typically used in robust controls and decompose it into a problem of finding the model uncertainty and controller uncertainty (see Section 4.1), a measurement of strong validation data versus weak validation data (see Section 4.1.2), projections of model uncertainty to areas not tested (see Section 4.1), and improved means of calculating a tolerance interval (see Appendix B).
Here, a dc-dc boost converter will be used to demonstrate the model and control algorithm validation process. Power converters are used throughout aircraft to help regulate voltage levels and impact power quality. With advances in electrical, mechanical, and structural design on-board more electric aircraft, the electrical systems have undergone considerably higher level of stresses, thus reducing power quality [6], [7]. Better designs of robust controllers for power converters that can handle the disturbances presented on the aircraft can improve power quality and lead to components with longer life expectancy. By taking advantage of the model validation approaches presented here, the testing of new control laws can be reduced, avoiding the typical trial and error methods.

One of the keys to any experimental or simulation test is the amount of testing performed. Taking too many data points will increase cost and time, while taking too few points means important information may not be found. Through experimental design [8], an optimized approach to testing is taken. Experimental design is used throughout this process with the goal of reducing the number of tests without losing information [9].

The remainder of this work is divided into nine chapters. The next chapter, Background, reviews literature in the areas of model validation, statistics, system identification, robust controls, and dc-dc boost converters. The following chapter, Problem Definition, offers clarification to the problems associated with model and control algorithm validation. The chapter Generalized Solution discusses an approach to decomposing the problem into separate problems of model validation and control algorithm validation. Chapter V, DC-DC Circuit Model Application, reviews the dc-dc model development, reviews the software setup, discusses the controller design, and presents a detailed process for model and control algorithm validation. The following chapter, Example Problems, presents several example problems to demonstrate the process and benefits of the key contributions. The following chapter, Hardware Design and Uncertainty Analysis, describes the
hardware used for the experiments and uncertainty associated with the components and measurement system. The next two chapters are Model Validation and Control Algorithm Validation, these chapters present the experimental results of the validation process with a dc-dc boost circuit developed on a printed circuit board. Finally, the last chapter includes conclusions and ideas for future research in this area.
CHAPTER II

BACKGROUND

The first section in this chapter, Model Validation, discusses literature in this area and explains a few strengths and weakness of the various techniques. System identification is discussed next, focusing on areas that will be used within this dissertation. Robust control is one of the topics of this proposal and work done in linear robust controls will be discussed. The next discussion will focus on the statistical tests, procedures and processes that will be used to assess the data. Included in this discussion will be design of experiments and how it can improve data collection optimization. Dynamic measurement uncertainty quantification is discussed next and its relation to the validation effort. Finally, a discussion on dc-dc boost converters will be given and various techniques that have been used in the past for control.

2.1 Model Validation

George P. Box wrote “essentially, all models are wrong, but some are useful” [10]. Which begs the question how useful? Through validation one can begin to understand how useful a model is. Model validation defined by MIL-STD 3022 is “the process of determining the degree to which a model, simulation, or federation of models and simulations, and their associated data are accurate representations of the real world from the perspective of the intended use(s)” (i.e., “did you build the right model”) [11]. The term model validation is typically well defined, but processes and tools
for model validation are still being developed. There are a number of standards that have discussed
how one might perform and/or document model validation [11, 12, 13, 14, 15, 16]. The references
[12] and [13] are from the American Society of Mechanical Engineers (ASME) and were developed
out of the computational solid mechanics group and primarily discuss means of applying p-box
type techniques (see Section 2.1.2) to the validation problem. The reference [14] is from ASME
computational fluid dynamics and heat transfer group and deals with the identification of parameters
of an item or experiment by examining all the sources of uncertainty. The reference [15] discusses
the idea of model validation in the context of computational fluid dynamics. The references [11] and
[16] are similar in the focus on the documentation that should be included in the model validation
process.

In [9, 13, 17, 18], the authors tend to refer to a system response quantity (SRQ) or quantity of
interest (QOI). The SRQ/QOI is the response that the user is most interested in measuring perfor-
mance of the system. This SRQ could be tip deflection of a beam [13], a function of voltage output
from the system [9], or could be a transient time measurement [17].

The remainder of this section discusses various techniques of model validation found in lit-
erature. The properties that make a good validation metric are discussed in [18] and [19]. The
property of note discussed here is an indication of statistical relevance of the comparison between
the physics modeled and the experimental results. Several of these approaches are useful and could
be considered for problems other than the dc-dc boost circuit.

2.1.1 Feature Examination

There are two general categories of feature examination, qualitative and quantitative. Qualitative
feature examination is graphical comparison between the simulation and the experimental data.
Unfortunately this tends to be one of the most used methods of “model validation”, but lacks an unbiased evaluation and lacks a statistical measure of performance.

The second method, quantitative feature examination, includes methods such as signal norms, normalized square error, and others as documented in [20]. Also, in [20] the authors develop a time based metric for measuring “difference” between time histories. Specifically the metric calculates magnitude, phase, and shape differences between signals then weights the measurements to arrive at a single metric. These methods at a minimum offer an unbiased evaluation of the experimental data when the method is chosen before data is obtained and a criteria for validation is pre-specified (i.e., the normalized square error should be less than a specified value). Unfortunately, these methods lack a statistical measure of performance.

2.1.2 P-Box Type Techniques

In the book [18] and articles [21], [22] the author discusses the various types of uncertainty that should be considered in validation, how one might propagate uncertainties in a model and how one might compare the output of a model and experimental results. The experimental comparison is done with a cumulative distribution area metric defined as

$$d(F, S_n) = \int_{-\infty}^{\infty} |F(x) - S_n(x)| dx,$$

where $F(x)$ is the cumulative distribution function (CDF) from the simulation and $S_n(x)$ is the CDF from the experiment. In [13] the area metric is normalized by the mean of the experimental outcomes to derive a percentage difference in simulation versus experimentation. With this method of validation, there is a lack of statistical significance to the test as presented (i.e., what Type I or Type II error would be expected from this test).
2.1.3 Mean Value Techniques

In [18] the author presents a method, mean comparison, that puts forth calculation of a confidence interval around the mean of the experimental data then sums the error from the simulation. The appropriate equation for the confidence interval is

\[
(y_m - \bar{y}_e - t_{\alpha/2,\nu}s/\sqrt{n}, y_m - \bar{y}_e + t_{\alpha/2,\nu}s/\sqrt{n}),
\]

where \( y_m \) is the SRQ from the model, \( \bar{y}_e \) is the average SRQ from the experimental data from \( n \) number of replications, \( s \) is the sample standard deviation from the SRQ, and \( t_{\alpha/2,\nu} \) is the student-t statistic where \( \alpha \) represents risk, and \( \nu \) the degrees of freedom. This technique is extended to multiple operating conditions and linear/nonlinear regression. The purpose of the mean value technique is not to predict how a system will behave but to identify a parameter of interest in the experimental process (thus the use of confidence intervals as opposed to tolerance intervals). This technique thus offers both an unbiased validation metric and a statistical significance. Unfortunately for control theory, the uncertainty calculation desired is not the mean value but a maximum value. The authors apply this methodology and use the maximum value of the error, but rely on replicated tests at the condition that causes the maximum condition. Under the approach presented here and use of tolerance intervals, the high number of replicated tests is not needed.

2.1.4 Bayesian Hypothesis Testing (interval based)

Another technique of interest found in [23, 24, 25] deal with Bayesian hypothesis testing (interval based). In this test, a comparison is made between the hypothesis \( H_0 : |\mu| \leq \epsilon \) and \( H_1 : |\mu| > \epsilon \), where \( \mu \) is the system response quantity or vector of system response quantities, and \( \epsilon \) is the interval of interest (i.e., the model error should be less than some fixed value). A Bayes factor is calculated as \( B_m = \frac{P(D|H_0)}{P(D|H_1)} \), where \( P(D|H_0) \) is the probability of the experimental data given \( H_0 \) is true, and \( P(D|H_1) \) is the probability of the experimental data given \( H_1 \) is true. If the Bayes factor is greater
than 1.0, then it is more likely that the model should be accepted then rejected. This approach thus offers both an unbiased evaluation, though some authors claim there is bias due to $\epsilon$ selection, and a statistical significance to the test. The need for use of the $\epsilon$ in the formulation of the problem is acceptable as validation requires an intended use for the model and specifying $\epsilon$ is part of that specification. One issue to this approach is the complexity of constructing the probabilities and thus will be examined in the future.

2.1.5 Whiteness Tests

Typically done within the field of system identification for model validation or sometimes called model diagnostics \[10\] are tests involving calculation of the auto-correlation and cross-correlation of the residuals and inputs. As suggested by \[10\] the equations for auto-correlation estimation are

\[
c_k = \frac{1}{N} \sum_{t=1}^{N-k} (z_t - \bar{z})(z_{t+k} - \bar{z}) \quad k = 0, 1, 2, \ldots, K,
\]

\[
r_k = \frac{c_k}{c_0}
\]

where $z_t$ is the discretization of the signal and a value at index $t$, $k$ is the lag under consideration, $K$ is the maximum lag being examined, $\bar{z}$ is the mean value of the signal, and $N$ are the number of data points. This formulation is slightly different than what is used in the field of signal processing where typically the mean values ($\bar{z}$) is not subtracted from the signal ($z_t$). Once the auto-correlation has been constructed it can be examined to determine if any one lag is significant that has not been modeled, or can be used as a whole to examine whether all lags are insignificant in affecting the model. Typically, vector $\tilde{Q}$ in Eq.(2.2) is calculated and checked to determine if it is likely chi-squared distributed with (K-p) degrees of freedom

\[
\tilde{Q} = n(n + 2) \sum_{k=1}^{K} (n - k)^{-1} r_k^2.
\]
Note, this formulation is modified from a form developed by Ljung-Box-Pierce [26]. Also note, it is recommended that the auto-correlation check be applied to the residuals of the model after an error model is constructed.

The whiteness test ensures that all signal content that can be used for a model is obtained. The method for model and control algorithm validation presented here avoids this consideration to ensure that models fit a form that a controller can be developed with. For example, this work assumes the model is in strict feedback form. To obtain a model in strict feedback form, the residuals between the control model and validated model may be auto-correlated, thus another method of validation is necessary. This metric though is unbiased and offers a statistical significance.

2.1.6 Anderson-Darling Tests

Two Anderson-Darling tests will be described here. The first test, the Anderson-Darling normality test [27], is a goodness of fit statistic that can determine the likelihood that a particular data set comes from a normal distribution. In this case, it can be used as a measure of the likelihood that the residuals are normally distributed and thus the model likely captures all signals of interest other than noise.

The second test, the Anderson-Darling K-sample test [28], tests whether two data sets are likely to come from the same distribution. The power in this test is that if both data sets are likely to come from the same distributions and one data set is produced by a model and the second from experimental data, then the model likely could have generated the data and is a good model of the data. This metric is unbiased and offers a statistical significance. The issue with these tests is that the model typically is required to match the data in a manner that does not allow for non-normalized error.
2.2 System Identification

Several ideas from system identification will be used to solve the problem of identification of uncertainty models and parameters. The system identification field typically considers either black-box modeling or white-box modeling, and recently has started to investigate gray-box modeling [29, 30]. Black-box modeling is characterized as modeling with little or no prior knowledge of the system in question. A class of models are considered (e.g., auto-regressive) and several analysis approaches are taken to determine the order of the model and parameter choices [31]. White-box modeling depends primarily on physics based modeling tools and processes. White-box modeling tuning involves selection of optimized parameters to minimize a cost function. Finally gray-box modeling techniques involve combining the two techniques in some manner though authors do not always agree on what constitutes a gray-box model. In some literature the white box modeling approached described where parameters are tuned is classified as a gray-box technique [32]. For this dissertation, gray-box modeling is defined as a combination of physics based model with a polynomial black-box model. The polynomial black-box model will be shown to have the ability to take into account parameter uncertainty and model form uncertainty for model and control algorithm validation.

The other aspect of system identification that will be used here is that of continuous time system identification as opposed to discrete system identification. Continuous system identification has the benefits of presenting results in the continuous time domain where most physics based modeling is derived [29, 31, 33]. Obviously there are means to convert from continuous to discrete and discrete to continuous, but there are many pitfalls in those approaches, and parameters do not always convert in a usable fashion [29]. Due to the inclusion of physics based modeling in the work presented here, continuous system identification is used to avoid the conversion of the physical modeling in the continuous domain to the discrete domain.
A major issue with continuous system identification is the calculation of the state derivatives. In [34] an overview of several methods of handling the derivative of the states is described. In the state variable filter approach [29], successive filters of the states are applied to the derivatives and all states. Consider a state equation

$$\frac{dx_1}{dt} = \theta_1 x_1(t) + \theta_2 x_2(t),$$

where $\theta_1$ and $\theta_2$ are unknown parameters and $x_1, x_2$ are states of the system. Using the $p$ operator, defined as $p^n x(t) = \frac{d^n x(t)}{dt^n}$, the above becomes

$$px_1(t) = \theta_1 x_1(t) + \theta_2 x_2(t),$$

then filtering both sides by a common filter,

$$\frac{\lambda p}{p + \lambda} x_1(t) = \theta_1 \frac{\lambda}{p + \lambda} x_1(t) + \theta_2 \frac{\lambda}{p + \lambda} x_2(t). \tag{2.3}$$

It is possible to implement the above filtering for discrete data without losing the benefit of the parameters being represented in the continuous time domain. Further, a linear least squares solution would exist so that $\theta_1$ and $\theta_2$ could be estimated. In this approach, information beyond the filter time constant is lost and the model is only good to that bandwidth, so careful choice of $\lambda$ should be done. Further, [29] states there are issues with state variable filter, specifically that the parameter estimates tend to be biased and inconsistent. Work done here does not disprove that statement, but does show this method works well and is relatively simple to implement. Further work should be done to examine other means to handle the state derivative issue to arrive at better parameter estimates.

### 2.3 Robust Controls

Classical robust controls [35, 36, 37, 38], guarantee stability by determining uncertainty bounds then designing an appropriate controller to obtain the needed stability. In [35, 39, 40, 41, 42] various
methods of identifying uncertainty or uncertainty bounds are described, typically by comparing a known model to experimental output. These works focus on linear systems and attribute the non-linearities of the systems in question to the uncertainties. These works then use a linear design approach to develop a controller for the system that will be stable for the uncertainties identified. In this dissertation, the control model uncertainty identification is done by examining the dynamics that are “left over” from performing some means of nonlinear control, in this case backstepping control. The approach here depends on a representation of the states and thus may not be globally applicable as some of the input-output approaches. However, by taking advantage of the state information, better understanding of the model errors can be obtained.

Nonlinear robust controls will be used throughout this dissertation including ideas from backstepping control, Lyapunov stability analysis, and nonlinear damping [43], [44], [45]. These areas will be discussed more in depth in Section 5.3.

2.4 Statistics

The statistics section is broken into five sections. The first describes the risks typically considered in statistics, namely of making producer errors and consumer errors. The second is a discussion on experimental design. Next, the diagnostic metrics of regression analysis are described. Next, a discussion on statistical intervals and their role in validation is given. Finally, an overview of different sampling techniques is given.

2.4.1 Risk

There are generically two types of risks that are used within the field of statistics. The first risk is the risk of a Type I error, or colloquially called producer risk. In the case of modeling, Type I error occurs when the modeler (“producer”) determines that a good model is not a good model. In this instance, more time is spent on improving a model that already meets the specified requirement. For
this work, the acceptable risk level of committing a Type I error will be specified as 5%. Throughout this dissertation it will be referred to as $\alpha$. The second risk that needs specified is that associated with Type II error, or otherwise known as “consumer” risk. A Type II error occurs when a “bad” model is said to be good. The end user (“consumer”) then uses the model assuming it is supplying answers within a certain risk level when the model is NOT performing to the level specified. This risk is associated with $\beta$ and will be set to a value of 1%. More formal discussions on Type I and Type II errors can be found in [8, 9, 46].

2.4.2 Experimental Designs

In [8] and [9] the authors discuss different experimental designs and benefits of these designs. Through experimental design [8], an optimized approach to testing is taken. Experimental design is used throughout this dissertation with the goal of reducing the number of tests without losing information [9]. For this work, two different experimental designs will be used. The first design used is a D-optimal design [8, 47] that will be used for tuning purposes. The goal of a D-optimal design is to minimize the determinant of $X'X$ where $X$ is the basis functions or regressor. The second type of design is based on using a Halton sequence [48]. It has been shown in [48] and [49] that quasi-random sequences provide faster convergence of statistical properties of a population than pseudo-random sequences. A design based on a Halton sequence will be used for tuning of several regression functions. It is pointed out in [50] that space filling designs can offer better means of tuning more complex functions. The primary reason for use of the Halton sequence is due to its random properties that are required for the acceptance sampling.

Finally, in the field of system identification and adaptive control, there are goals of persistency of excitation. A vector is persistently exciting if $X'X$ is positive definite [51] which equates to the eigenvalues being greater than zero. It is then obvious to see the relationship between the D-optimal designs and persistency of excitation.
2.4.3 Sampling

One of the keys to any experimental or simulation test is the amount of testing performed. Taking too many data points will increase cost and time, while taking too few points means important information may not be found.

Two areas of sampling will be examined. The first area involves tuning and training a polynomial model. For this area, use of the work done by DeLoach in [52] will be examined. In DeLoach’s work he develops equations for the number of samples required to meet Type I and Type II errors. Through his formulation, the user can choose to use required values or the best possible values. DeLoach recommends use of the best possible value, since most end users have difficulty specifying required values and/or do not have all the information (i.e., the process standard deviation). For the generic case, allowing the end user to specify a difference to detect and the estimated variance of the model the following formula can be used

\[
N = P(Z_\alpha + Z_\beta)^2 \frac{\sigma^2}{\delta^2},
\]

\[
P = \frac{(d + k)!}{d!k!},
\]

where \(\sigma\) is the ordinary standard deviation in the process, \(\delta\) is the difference to detect or maximum mean error allowed in response. Otherwise the least significant difference and the following sampling plan can be used

\[
\delta = 2\sqrt{2}\sigma,
\]

\[
N = \frac{1}{2} P \left( \frac{Z_\alpha + Z_\beta}{2} \right)^2.
\]

In [53] the author suggests to confirm a model by constructing a prediction interval for the model and counting the number of times new data exceeds the prediction interval (i.e., model was not able to predict performance at that point). The number of counts is compared to a critical number based
on a binomial test. A similar approach is taken here with a few caveats. First, instead of a prediction
interval a tolerance interval will be utilized. The second difference is in the confirmation test. This
dissertation uses as single-sampling plan from statistical quality control [46], [54], [55], and [56]
that ensure Type I and Type II risk levels are met.

The acceptance sampling technique is based on meeting the following two probabilities

\[
P_a(p_1) \geq 1 - \alpha, \\
P_a(p_2) \leq \beta,
\]

where \( p_1 \) is called the acceptable quality level (AQL), and \( p_2 \) is the rejectable quality level (RQL)
or lot tolerance percent defective (LTPD). In words, the sampling plan ensures the probability of
acceptance at the AQL level is higher than \( 1 - \alpha \), and lower than \( \beta \) at RQL. The binomial distribution
is used here as the population of possible computations (i.e., simulations) is large and effectively
samples can be pulled with replacement. These probabilities reference the binomial distribution
such that

\[
P_a(p) = \sum_{d=0}^{c} \left\{ \frac{n!}{d!(n-d)!} p^d (1-p)^{n-d} \right\}.
\]

The two probabilities can then be optimized over number of test runs (n) and number of allowed
bad points/rejects (c) (see [46] for more information on the optimization).

The confirmation tests, the number of tests executed, and the critical number not only protect
us from not identifying a good model but also identifying a bad model as good. Each confirmation
test is randomly chosen (i.e., the operating conditions are randomly chosen) and evaluated against a
pre-determined metric. For example, each confirmation test may measure the amount of error in a
given output and if that error exceeds some threshold then that individual confirmation test fails. If
the number of confirmation test fails exceeds the critical number than the validation test fails.
Note, the acceptance sampling plan as specified here is not a uniformly most powerful test because the hypothesis are not composite [57]. For this work, the null hypothesis for this test is $H_0: p < p_1$ and the alternative is $H_a: p > p_2$. Had the alternative been stated as $H_a: p \geq p_1$, and beta had been allowed to change this test would have been a uniformly most powerful test.

2.4.4 Regression Analysis

Regression analysis is a general methodology to understand relationships between dependent and independent variables [9, 50, 58, 59]. Linear least squares will be used to determine the relationships, and then diagnostic regression analysis will be performed on the results to ensure the model and parameters used are reasonable. A number of statistical measures including coefficient of determination ($R^2$), adjusted coefficient of determination ($R^2_{adj}$), coefficient of determination for prediction ($R^2_{pred}$), and the sample standard deviation ($s$) will be used to measure that reasonableness. Those metrics are defined as

$$s = \sqrt{\frac{1}{N-1} \sum_{i=1}^{N} (y_i - \bar{y}) \text{ where } \bar{y} = \frac{1}{N} \sum_{i=1}^{N} y_i},$$

$$SS_{err} = \sum_{i=1}^{N} (\hat{y}_i - y_i)^2,$$

$$SS_{tot} = \sum_{i=1}^{N} (y_i - \bar{y})^2,$$

$$R^2 = 1 - \frac{SS_{err}}{SS_{tot}},$$

where $N$ is the number of samples in the test and $\bar{y}$ is the value of $y$ as predicted by the model. The coefficient of determination, $R^2$, is an indicator of the amount of the variation that is explained by the model [9]. The adjusted coefficient of determination, $R^2_{adj}$, further indicates “goodness” of a model by discounting $R^2$ as less variation is explained by more terms in the model. Mathematically

$$R^2_{adj} = 1 - \frac{SS_{err}/(N - p)}{SS_{tot}/(N - 1)},$$
where $p$ is the number of regression coefficients in the model. The final indicator is the coefficient of determination prediction, $R_{pred}^2$, value which is an indicator of how well the model will work in the prediction space [58]. Mathematically

$$R_{pred}^2 = 1 - \frac{PRESS}{SS_{tot}},$$

where PRESS is the prediction error sum of squares and uses the “leave-one-out” method of cross validation [59].

$$PRESS = \sum_{i=1}^{N} (y_i - \hat{y}_{(i)})^2,$$

where $\hat{y}_{(i)}$ is the prediction of the model without the $i^{th}$ point for model construction. Rather than determination of each separate model, one can use the hat matrix $H$ with elements $h_{ij}$.

$$PRESS = \sum_{i=1}^{N} \left( \frac{y_i - \hat{y}_{i}}{1 - h_{ii}} \right)^2,$$

$$H = X(X'X)^{-1}X',$$

where $X$ is the regressor.

### 2.4.5 Statistical Intervals

Within the field of statistics the three intervals of interest are confidence intervals, prediction intervals, and tolerance intervals [8]. In this work, tolerance intervals are of greatest interest as tolerance intervals can be used to “predict” the likely behavior of the remaining population (i.e., how the model will behave at similar points not tested). Prediction intervals typically look towards predicting the value of a single next test. Confidence intervals are used for specifying the accuracy of a prediction of a parameters of the population.

A tolerance interval indicates the percentage of the population that will exhibit a characteristic with a certain level of confidence. An upper single-sided tolerance interval for most distributions is specified as $\bar{y} + ks$ where $s$ is the sampled population’s standard deviation of the characteristic,
\( \bar{y} \) represents the sampled population’s mean value, and \( k \) is a factor dependent on the confidence level, population percentage [58], and distribution type. A one-sided calculation of \( k \) for normally distributed data is

\[
k = \frac{t_{\alpha,N-1,\phi}}{\sqrt{N}},
\]

\[
\phi = z_{p_1} \sqrt{N},
\]

where \( t_{\alpha,N-1,\phi} \) is the inverse to the non-central T cumulative distribution function, \( N \) is the number of samples, and \( z_{p_1} \) is the inverse normal distribution with probability \( p_1 \), \( \alpha \) is the probability that \( p_1 \) of the population will exhibit the characteristic.

As mentioned, the \( k \) value used in the tolerance interval calculation in Eq. (2.6) assumes a normal distribution of the population, so this interval calculation should be used with care. A distribution-free, or non-parametric, tolerance interval calculation is discussed in [60] and [61]. An upper single sided tolerance interval is

\[
P(F(U) \geq p_1) \geq \alpha,
\]

where \( F(U) \) represents the cumulative distribution function at limit \( U \), and \( \alpha \) is the probability that \( p_1 \) of the population will exhibit the characteristic. The non-parametric solution to the one-sided problem is to order the sequence of data to be examined such that \( x(j) \leq x(j+1) \) for \( j = 1 : N - 1 \) where \( N \) is the number of elements in the data set. An upper limit for the tolerance interval is found such that \( U = x(s) \). The beta distribution is used to solve for \( s \) such that

\[
1 - F_\beta(p_1, a, b) \geq \alpha
\]

where \( F_\beta \) is the cumulative beta distribution, such that

\[
F_\beta(x|a, b) = \frac{1}{B(a, b)} \int_0^x t^{a-1}(1 - t)^{b-1}dt
\]
where $B(a, b)$ is the beta function \cite{60}. The goal is to find the smallest value of index $s$ of the ordered data series where $a$ and $b$ in Eq.\((2.7)\) are

\[
a = s,
\]

\[
b = N - s + 1.
\]

### 2.5 Dynamic Measurement Uncertainty Quantification and Propagation

To perform model validation there is a need to understand the uncertainty in the measurement system and components used in the experiment. Without this understanding, one might assign the uncertainty to the modeling effort as opposed to the measurement system or component variability. The works \cite{17, 62, 63} apply these ideas to dynamical systems.

In \cite{17} the author discusses dynamic uncertainty quantification, sensitivity analysis and uncertainty propagation. In the work, the author utilizes a simple RLC circuit to measure the uncertainty of the components and the input voltage. Using a model, these uncertainties are propagated to an output voltage and compared against experimental data. The author uses the Anderson-Darling K-sample test mentioned above to show validation, but the results showed an invalid model as it did not pass the test the author proposed. Although the model the author developed did not pass the proposed validation test, it was sufficient for some types of analysis, one goal of the work here is to state how good the model is rather than just a simple valid/invalid test.

Another area that was discussed within \cite{17} but discussed more in \cite{62} and \cite{63} is that of dynamic measurement uncertainty quantification. Dynamic-Measurement Uncertainty Quantification’s (D-MUQ) technical definition is, “The probabilistic temporal error between an assigned physical property or characteristic and its actual value”. In layman’s terms it is the identification of the time-based error between perceived and actual data. The difference between D-MUQ and standard means of measurement uncertainty quantification associated with sensor data is the dynamic
characterization that is incorporated in D-MUQ. Typical measurement uncertainty, the status quo, is developed using static measurements, but validation of dynamic models, analysis of control systems and documentation of dynamic system signals are temporal in nature, not static. Therefore, to obtain a better understanding of the time based errors in complex, integrated and highly interactive systems, D-MUQ is needed. The status quo method will result in inaccurate measurement uncertainty quantification when applied to dynamic systems. The inaccuracy leads to additional risk which has ramifications during decision making events. In [62] and [63] dynamic measurement uncertainty is addressed for voltage data acquisition. In this paper, the authors discuss in-situ testing of a voltage measurement device in regards to not only static uncertainty but temporal uncertainty.

2.6 DC-DC Boost Converter Model

DC-DC boost converters efficiently convert a lower level unregulated voltage signal to a stabilized higher level voltage signal [64]. The typical dc-dc boost converter has one active switch and a passive switch (diode) (see Fig. 2.1). The inductor in the circuit stores energy in a magnetic field while the active switch (n-MOSFET) is closed. The passive switch (diode) will be closed and electrical energy stored in the capacitor supplies the output voltage and current for the load. When the active switch is open (passive switch will close), the inductor supplies the power to re-charge the capacitor and supplies power to the load.

Traditionally control of dc-dc boost converters has been done with passive components in either a voltage only control or an inner current loop control with an outer loop voltage control [64] [65]. These controllers are historically developed around a linearization of the boost circuit. For voltage control, the duty cycle of the active switch is adjusted based on error in a reference signal and the output voltage. A longer duty cycle will tend to increase the voltage at the output. When the current loop is added, an inner current loop closes the active switch when the inductor current exceeds a
reference current supplied by an outer loop voltage controller. This method tends to protect the components from over-current and improves the stability of the system.

With the linearized equations of a dc-dc converter, lead-lag, PI, or PID control law development can be done to ensure phase margin present in the circuit is adequate for stability [64]. For these
type of controllers, the following small signal analysis is used\textsuperscript{[64]}

\[ T_p(s) = \frac{v_o(s)}{d(s)}, \]  
\[ T_p(s) = T_{px}\frac{(s + w_{zn})(s - w_{zp})}{s^2 + 2\zeta w_o s + w_o^2}, \]  
\[ T_{px} = -\frac{V_o - r_c}{1 - D R_L + r_c}, \]  
\[ w_o = \sqrt{(1 - D)^2 R_L + r_L C}, \]  
\[ \zeta = \frac{L + C[r(R_L + r_c) + (1 - D)^2 R_L r_C]}{2\sqrt{L C(R_L + r_c)((1 - D)^2 R_L + r_L C)}}, \]  
\[ w_{zn} = \frac{1}{C r_C}, \]  
\[ w_{zp} = \frac{R_L(1 - D)^2 - r}{L}, \]  
\[ r = r_l + D r_{ds} - R_F (D - 1), \]

where \( v_o(s) \) is the small signal output voltage, \( d(s) \) is the small signal duty cycle command, \( D \) is the steady state duty cycle command, \( R_L \) is the load resistance, \( C \) and \( L \) are the capacitance and inductance in the circuit, \( r \) is a combination of the series resistance of the inductor (\( r_l \)) plus the forward on resistance of the diode (\( R_F \)) and the on resistance of the n-MOSFET (\( r_{ds} \)), and \( r_C \) is the series resistance of the capacitor. The outer loop voltage controller then can be designed using \( T_p(s) \).

These linear control laws though do not consider large signal stability. In\textsuperscript{[66]} the authors demonstrate the importance of examining large signal analysis as some small signal designs can lead to instabilities (these are typically voltage control only). Several authors have investigated large-signal analysis of the boost circuit and checked stability based on Lyapunov’s direct method\textsuperscript{[67]}. The analysis in\textsuperscript{[67]} used a lead lag controller from passive electrical components for control and set the gains to ensure stability. The analysis only included losses from the inductor. The author however assumed the output voltage was equal to the voltage across an ideal capacitor and did not include the series resistance of the capacitor in the analysis (left plane zero \( w_{zn} \)).
In [68] and [69] feedback linearization techniques are considered for control. In both cases, only ideal circuit elements are considered. In [68] the author shows that for an ideal boost converter that feedback linearizing control using direct voltage control will have unstable zero dynamics of the inductor current and that a feedback linearizing controller using inductor current as an inner loop with an outer loop voltage controller is stable. In this formulation the authors assume a fixed load. In [69] the author shows similar results for an inner loop current control and outer loop voltage control, but extends the results for varying loads.

A number of fuzzy control laws have been developed for power converters. In [70] the author designs a PI, PID, fuzzy control (PD portion was fuzzified with an integrator outside of the fuzzy system), and a sliding mode fuzzy controller where the gains for the fuzzy system were determined by a first order switching model. All controllers developed in the work were based on controlling voltage. In [70] the author makes a comment that the right half plane zero, \( w_{zp} \) in Eq.(2.8), makes control design very difficult, further indication of the need to include the losses (\( r \)) in the large signal analysis.

In [71] a sliding mode controller was developed for a boost circuit. In evaluation of the control law developed the author tested the control against changes in reference voltage command, step load changes, and input voltage changes. The models simulated did not include \( r \) nor the series resistance of the capacitor from Eq.(2.8).

In [72] the author makes use of model predictive control for parallelized buck converters. Model predictive control is an area of research that can greatly benefit from model validation as mentioned in [72].

In [73] and [74] backstepping control is performed on an ideal boost circuit similar to the one used by [68]. In [73] an adaptive and non-adaptive backstepping approach are designed and tested in simulation with an unknown / varying load resistance. In [74] the authors consider parameter
uncertainty in the inductor, capacitor, resistor, and input voltage but do not consider model form un-
certainty which is one focus of this dissertation. In [74] the authors make use of nonlinear damping
to handle uncertainty and find a boundary condition, which is dependent on the variation, whereas
in this dissertation the bound will be fully dependent on the parameters chosen for the control law
while the control law is dependent on the uncertainty bounds.
CHAPTER III

PROBLEM DEFINITION

Model or control validation is either not done or not done with statistical measures for a number of systems, subsystems, or components. In some instances, model validation is claimed after single traces of simulation data are overlaid on experimental data with little or no information as to the processes involved in making these conclusions (i.e., qualitative feature extraction). As such the reliability of these models is unknown and can only be used with great risk in making decisions. Further, using a non-validated model for control development can lead to non-optimal control designs or the typical trial and error approach to control development. The problem that will be addressed here is to identify a process of model and control algorithm validation using statistical measures that offer the user a measurement of risk associated with using the model.

Power converters are used throughout aircraft to help regulate voltage levels and have the potential to greatly impact power quality. With increased electrification of aircraft, the loads on the electrical system have grown and continue to grow. There is a need to improve the power quality on the aircraft and predict when power quality will be an issue \[2\], \[6\]. A more robust control that can mitigate the disturbances presented on the aircraft can enable the ability to improve power quality resulting in components with longer life expectancy. Application of a model validation approach can reduce the amount of the trial and error testing of the power converter control law saving time and money.
To develop a more robust control for a problem like the power converter, three issues are likely to emerge that will need to be addressed by any model and control algorithm validation process. The first issue is identification of a system response quantity that will be used in the statistical hypothesis formulation. In non-linear robust control development, the system response quantity is the total uncertainty or a limit of the total uncertainty in the state equation \[ \dot{x} = f(x) + g(x, u) + \delta(x, u) \]

where \( x \) is the state of the system, \( f(x) \) is the known unforced state dynamics, \( g(x, u) \) is the known forced state dynamics, and \( \delta(x, u) \) is the unknown forced and unforced state dynamics (i.e., total uncertainty). Thus a model and control validation process should address identification of \( \delta(x, u) \) or at a minimum a limit on the magnitude of \( \delta(x, u) \).

Second, certain model forms may not be conducive to the user needs, especially in control. As such, there is a need to develop a model validation methodology that does not rely on “whiteness” of the residuals (i.e., \( \delta(x, u) \) will likely not be normally distributed or could be cross-correlated with the input). For example, a considerable amount of control theory is developed around linear systems. As such, if one wants to develop a linear control for a plant model, typically that model should be linear or at least the model will need to be linearized. The errors between the linear or linearized model and experimental data may or may not be normally distributed or the errors might be cross-correlated to the inputs and thus techniques within the system identification fields are difficult or impossible to use to evaluate a model. Expanding the allowed control methodologies to include non-linear control does not solve the issue, though the restrictions are considerably relaxed over linear control methodologies. Thus, the process for model validation and control validation should use tools that are independent of the distribution of the error between the model and experimental data.
Last, an issue faced in model validation is the ability to test in the application domain \[18\]. The application domain refers to the boundary conditions or environment that the component, subsystem, or system will encounter when in use. Often those boundary conditions may not be able to be replicated in the lab, so there is a need to understand the impact or have the ability to predict performance in the application domain. Further, for the problem of control development, ideally a model would be validated first and then the control development would begin. Once the control development is complete, the limit on the model uncertainty may be higher due to the state trajectory that may occur with the new control laws (i.e., the state trajectory may be in a domain where the model was not evaluated). Thus, a validation process will need to predict the impacts of the new state trajectory on the model form uncertainty thus predicting the model uncertainty in the application domain.
CHAPTER IV

GENERALIZED SOLUTION

This chapter will discuss the generalized solution for total uncertainty estimation needed for robust control development by presenting a means to decompose the problem into one of model uncertainty estimation and control uncertainty estimation. At the end of the chapter, the different types of simulations used in the analysis will be discussed.

4.1 Model and Control Uncertainty Analysis

The generalized solution to the model and control uncertainty analysis starts with a description of the assumptions that will be used throughout the dissertation, then discusses how the estimation of future uncertainty will be performed, along with different estimations that will be used for comparison.

4.1.1 Uncertainty Development

The class of systems to be considered for control are ones in strict feedback form \[45\] (though other forms could be used). In this form, the differential equations are in a form such that each subsystem (i.e., first state equation) can be solved before including the second state equations and so on. The following system is demonstrative of strict feedback form (it includes the model uncertainty
\[
\dot{x}_1 = f_1(x_1) + g_1(x_1)x_2 + \delta_1(X, P, s, u) \tag{4.1}
\]
\[
\dot{x}_2 = f_2(x_1, x_2) + g_2(x_1, x_2)u + \delta_2(X, P, s, u). \tag{4.2}
\]

In the equation above, \( X = [x_1, x_2]^T \in \mathbb{R}^2 \). The uncertainty functions, \( \delta_1 \) and \( \delta_2 \), can be functions of parameters \( (P) \), states \( (X) \), disturbances \( (s) \), or input \( (u) \). The uncertainties can be time varying functions though constant uncertainties are considerably easier to measure as will be discussed shortly. It is assumed for this work that the disturbance \( s \) is known and differentiable. For simplicity dependencies on \( X, P, s, \) and \( u \) are replaced with \( \Phi_{ji} \). Define a domain \( G \) that represents the boundary conditions \( (s, u) \), the states of the plant, and the parameters \( P \) and thus \( \Phi_{ji} \in G \) where \( i \) corresponds to the state, and \( j \) will be defined below. The functions \( f_1, g_1, f_2, g_2, \delta_1, \) and \( \delta_2 \) are required to be differentiable and Lipschitz within domain, \( G \), for purposes of system identification and uncertainty estimation. It is assumed that the Lipschitz constant is consistent throughout \( G \) (i.e., if one calculates the Lipschitz constant over a subset of \( G \) it will be similar to other subsets of \( G \)).

The functions \( f_1, g_1, f_2, \) and \( g_2 \) are known dynamics of the plant and will be used for control. For uses within the backstepping control the functions \( f_1, g_1, f_2, g_2, \delta_1, \) and \( \delta_2 \) are required to also be sufficiently smooth \cite{45}. For purposes of identification of the uncertainty, there is no requirement on the boundedness of the functions. For control purposes this assumes that \( |\delta_1(X, P, s, u)| \leq \Delta_1 \) and \( |\delta_2(X, P, s, u)| \leq \Delta_2 \) within \( G \) where \( \Delta_1 \) and \( \Delta_2 \) are constant, but unknown. These unknown constants will be found through identification of models of the uncertainties. The requirement of boundedness and domain can be relaxed, but a different control modification would then be needed.

See \cite{45} or \cite{44} for approaches to handle unbounded uncertainties.
To determine the total uncertainty in each of the above state equations, simple algebraic manipulation is used to find

\[ \delta_1(\Phi_{j1}) = -f_1 - g_1 x_2 + \dot{x}_1 \equiv \dot{x}_1 - \dot{x}_{c1}, \tag{4.3} \]

\[ \delta_2(\Phi_{j2}) = -f_2 - g_2 u + \dot{x}_2 \equiv \dot{x}_2 - \dot{x}_{c2}, \tag{4.4} \]

where \( \dot{x}_{ci} \) will be known as the control differential. The control differential is used in the development of the controller (i.e., \( \dot{x}_{c1} = f_1 + g_1 x_2 \) and \( \dot{x}_{c2} = f_2 + g_2 u \)).

With the definition of uncertainty determined, there is a desire to de-compose the problem to make estimation of the uncertainties \( \delta_1 \) and \( \delta_2 \) tractable. It will be shown that the total uncertainty limit is equal to the model uncertainty limit added to the control uncertainty limit. To start, within domain \( G \), exist data sets \( D_j \) over which the uncertainty estimations are to be performed.

- \( D_t \) is an experimental data set used for tuning and system identification. The initial use for the data set is system identification of the model, then it will be used to identify the model uncertainty, and controller uncertainty. The controller uncertainty here will be used to construct an augmented controller if necessary as described in Section 5.3.3.

- \( D_{sv} \) is a simulated data set of the intended validation set. It will be used in the prediction of the data set \( D_v \).

- \( D_v \) is an experimental data set used for model validation. The inputs from the data set will be used to create a simulated data set \( D_s \). Finally this data set will be used to update the model uncertainty for the future data set and calculate a control uncertainty based on the augmented control laws.

- \( D_s \) is a simulated data set using the augmented control. The data set is constructed from the validation data set \( D_v \) using the non-controller inputs and disturbances and modifies the control signal \( u \) per the augmented controller. This data set will be used for control validation.
• $D_f$ is the experimental future data set to be taken. This data uses the modified control and similar inputs as the validation data set. The goal of this work will be to predict behavior of this data set. Validation of the model and control and their corresponding uncertainty limits will occur with data sets $D_v$ and $D_s$.

It is important that the validation data set, $D_v$, is different from the tuning data set, $D_t$. A larger difference between the two data sets increases the likelihood one will have robust validation. In [18], Oberkampf discusses the need for evaluating a tuned model on data quite different then what was used for tuning. One item proposed here is a metric to differentiate between strong validation and weak validation data sets to ensure robust validation. The metric will be presented in section 4.1.2.

Before beginning the determination of the control and model uncertainty a few definitions need to be established. For this work the signal norm will be defined as

$$||z||_{\infty} = \sup_{t \geq 0} ||z(t)||$$

The vector norm used is a 2-norm,

$$||Z||_2 = \sqrt{|Z|^2}.$$

The following describes how one might de-compose the problem so that model uncertainty and control uncertainty can be calculated separately and then combined to find the total uncertainty.

$$\delta_i(\Phi_{ji}, t) = \dot{x}_i(\Phi_{ji}, t) - \dot{x}_{ci}(\Phi_{ji}, t) \text{ Total uncertainty,}$$

$$\delta_{mi}(\Phi_{ji}, t) \equiv \dot{x}_i(\Phi_{ji}, t) - \dot{x}_{mi}(\Phi_{ji}, t) \text{ Model uncertainty,}$$

$$\delta_{ci}(\Phi_{ji}, t) \equiv \dot{x}_{mi}(\Phi_{ji}, t) - \dot{x}_{ci}(\Phi_{ji}, t) \text{ Control uncertainty,}$$

$$\delta_i(\Phi_{ji}, t) = \delta_{mi}(\Phi_{ji}, t) + \delta_{ci}(\Phi_{ji}, t) \text{ Total uncertainty} = \text{Model uncertainty + Control uncertainty}$$

where $x_{mi}$ is the model differential. In cases where the model differential is in strict feedback form, the controller uncertainty will be zero. However, in cases such as the dc-dc boost circuit problem
here that may not be the case. Assume that the following bounds exist over a data set $D_j$.

$$\Delta_{mi}(D_j) \geq ||\delta_{mi}(\Phi_{ji}, t)||_\infty \forall \Phi_{ji} \in D_j \subset G,$$

$$\Delta_{ci}(D_j) \geq ||\delta_{ci}(\Phi_{ji}, t)||_\infty \forall \Phi_{ji} \in D_j \subset G,$$

$$\Delta_i(D_j) \equiv \Delta_{mi}(D_j) + \Delta_{ci}(D_j) \geq ||\delta_i(\Phi_{ji}, t)|| \forall \Phi_{ji} \in D_j \subset G.$$

To find an upper limit on $||\delta_i(\Phi_{fi}, t)||$ one needs to estimate $\Delta_{mi}(\Phi_{fi})$ and $\Delta_{ci}(\Phi_{fi})$. Since the data set $D_f$ is a future data set, it will need to be estimated. The following is a method for calculating those variables. First by using the negative triangle inequality, as given in Appendix C.1, one obtains the following

$$||\delta_{mi}(\Phi_{fi})|| - ||\delta_{mi}(\Phi_{vi})|| \leq ||\delta_{mi}(\Phi_{fi}) - \delta_{mi}(\Phi_{vi})||$$

$$||\delta_{mi}(\Phi_{fi})|| \leq ||\delta_{mi}(\Phi_{fi}) - \delta_{mi}(\Phi_{vi})|| + ||\delta_{mi}(\Phi_{vi})||$$ (4.5)

Because of the Lipschitz property of the uncertainty, one can then show the first element in Eq. (4.5) is

$$||\delta_{mi}(\Phi_{fi}) - \delta_{mi}(\Phi_{vi})|| \leq L_{mi}||\Phi_{fi} - \Phi_{vi}||.$$ 

To calculate $L_{mi}||\Phi_{fi} - \Phi_{vi}||$ it will be assumed that $||\Phi_{fi} - \Phi_{vi}|| \leq ||\Phi_{vi} - \Phi_{ti}||$. Why can this assumption be made? This assumption is based on the fact the validation data is more similar to the future data set then to the tuning data set, and thus the “distance” is shorter between the validation to the future data set then the validation to the tuning data sets. The more different the validation data set is from the tuning data set, the more likely this assumption will hold. In this work, the goal is to predict the performance of the future data set $D_f$ (i.e., design a controller that will meet the performance requirements). The modeler can thus pick the validation data set first (i.e., so that it is similar to the future data) and then pick the tuning data set (i.e., different than the validation data), and perform the necessary validation and tuning to inform the control developer as to the
model uncertainty and model form. That may not always be possible (e.g., the future data set is at operating conditions that cannot be emulated in the lab and can only be created during flight). In the situation where the validation data set is different than the future data set, the future data set can be treated as another validation data set and the ideas that will be discussed in Sections 4.1.2 and 5.5.3 can be applied.

A similar property can be associated with the control uncertainty such that

\[ ||\delta_{ci}(\Phi_{fi})|| \leq L_{ci}||\Phi_{fi} - \Phi_{si}|| + ||\delta_{ci}(\Phi_{si})||. \]

It is assumed that \( ||\Phi_{si} - \Phi_{ti}|| \geq ||\Phi_{fi} - \Phi_{si}|| \) for control purposes. Again it is assumed that the distance from data set \( D_s \) to \( D_t \) is larger than \( D_f \) to \( D_s \). So that

\[
\delta_i(\Phi_{fi}, t) = \delta_{mi}(\Phi_{fi}, t) + \delta_{ci}(\Phi_{fi}, t),
\]

\[
\delta_i(\Phi_{fi}, t) \leq L_{mi}||\Phi_{vi} - \Phi_{ti}|| + ||\delta_{mi}(\Phi_{vi}, t)|| + L_{ci}||\Phi_{si} - \Phi_{ti}|| + ||\delta_{ci}(\Phi_{si}, t)||,
\]

\[
\delta_i(\Phi_{fi}, t) \leq L_{mi}||\Phi_{vi} - \Phi_{ti}|| + \Delta_{mi}(\Phi_{vi}) + L_{ci}||\Phi_{si} - \Phi_{ti}|| + \Delta_{ci}(\Phi_{si}). \quad (4.6)
\]

A simple example was created to demonstrate the prediction using the Lipschitz constant (see Fig. 4.1). It is assumed that tuning was done from \( x = 0 \) to \( x = 5 \) and that there is a desire to predict the uncertainty at \( x = 10 \). For this example, set \( X_1 = [0, 1, 2, 3, 4, 5] \) and \( X_2 = 10 \) and \( f(x) = \sin(\pi x) + 5x \). It will also be assumed that the Lipschitz constant can be determined exactly and for this function it is \( 5 + \pi \). The distance \( ||X_2 - X_1|| \) is 5, the Lipschitz constant is 8.14, the maximum uncertainty in the tuning range is 25, for a projected uncertainty limit of 65.7. The actual uncertainty is 50, and thus the Lipschitz estimate serves as an upper bound on the uncertainty.

The next two sections will discuss means to determine the four parameters of this equation. The Lipschitz constant and distance, \( L_{mi}||\Phi_{vi} - \Phi_{vi}|| \) and \( L_{ci}||\Phi_{si} - \Phi_{ti}|| \), will be discussed in Section 4.1.2 and the uncertainty limits, \( \Delta_{mi}(\Phi_{vi}) \) and \( \Delta_{ci}(\Phi_{si}) \), will be discussed in Section 4.1.3.
Figure 4.1: Example of using the Lipschitz prediction.
4.1.2 Lipschitz Constant Estimation and Distance Evaluations

In [75] an estimate of the Lipschitz condition was derived for a univariate black box problem. The authors stated “We remark that we have converted the Lipschitz constant estimation problem into a routine curve fitting problem.” There has been some work in [76] on Lipschitz estimation of multi-variate problems that required more points to find an estimate than what is used here. Here, a similar approach is taken as in [75] but for a multivariate problem, and making use of a polynomial based function estimate. Obviously, more complicated basis functions could be used, but issues of over-fitting of models can be problematic. The other key take-away from this section is that the desire is to calculate $L||x - y||$ and not just the Lipschitz constant. Because of this distinction it will be demonstrated it is important to normalize the problem, so that overestimation of the entire quantity likely does not occur.

The first step is to estimate the uncertainty function $\delta$ for the existing data sets (see Appendix E for a discussion on how to perform regression analysis for this problem). Note, the uncertainty function $\delta$ here can either be a model or control uncertainty function. This estimate at times can be good, but typically is limited over the region on which it is estimated and will be discussed in Section 4.1.4. Once an estimate of $\delta$ is created, the calculation of the partial derivative of the function could be performed to estimate the Lipschitz constant.

Rather than calculating the Lipschitz constant directly from this equation, the set of equations will be transformed to be of the form $\delta(Z_a) = z_{1a} + z_{2a} + ... + z_{na}$ where each $z_{ia}$ is a placeholder for each polynomial and parameter. For example, if the uncertainty equation was of the form

$$\delta(X_a) = a_1x_{1a} + a_2x_{2a} + ... + a_nx_{na}$$
then the transformation would be

\[ z_{1a} = a_1 x_{1a}, \]
\[ z_{2a} = a_2 x_{2a}, \]
\[ \ldots \]
\[ z_{na} = a_n x_{na}. \]

With the above transformation, the Lipschitz constant, \( L \), is \( \sqrt{n} \), and the distance component, \( \|X_1 - X_2\| \), becomes \( \|Z_1 - Z_2\| \) where \( Z_1 = [z_{11}, z_{21}, \ldots, z_{n1}]^T \) and \( Z_2 = [z_{12}, z_{22}, \ldots, z_{n2}]^T \).

Although, by this transformation it is not guaranteed that the answer will be less conservative, typically it has been. For a two state system, (i.e., \( n = 2 \)), early work showed the value for \( L\|X_1 - X_2\| \) to be quite large, and it was primarily due to \( a_1 \gg a_2 \) and \( x_{2a} \gg x_{1a} \). Although this is not guaranteed and in the case where \( a_1 \gg a_2 \) and \( x_{1a} \gg x_{2a} \) this transformation may overestimate \( L\|X_1 - X_2\| \) by \( \sqrt{2} \).

Further, this method removes the step of having to perform another optimization problem in the case where the function \( \delta(X_a) \) is higher than order one. If \( \delta(X_a) \) is higher than order one, then the Lipschitz constant becomes a function of \( X_a \) and an optimization is needed to determine the largest value of \( L \).

As mentioned earlier there was a desire to determine weak versus strong validation data sets. The distance calculation of the transformed system \( \|Z_1 - Z_2\| \) presented above is one such comparison. The distance function used throughout calculates the maximum distance over all the points in the first data set between its closest neighbor in the second data set by the euclidean distance function. Mathematically,

\[ \|Z_1 - Z_2\| = \max_j \left[ \min_i \left( (Z_1(j) - Z_2(i)) \cdot (Z_1(j) - Z_2(i)) \right) \right] \quad (4.7) \]
Figure 4.2 depicts what is considered relatively weak validation data, where the tuning data set and the validation data set is intermixed. Figure 4.3 demonstrates the notion of a strong validation data as the tuning data set is “farther” away from the validation data set then in the previous figure (i.e., distance of 0.5240 compared to 0.0775). The distance difference between the two sets is more than an order of magnitude. Ideally, a strong validation data set would encompass the area of intended use. A strong validation data set will be defined such that

\[ ||\Phi_{vi} - \Phi_{ti}|| \gg ||\Phi_{vi} - \Phi_{svi}||. \]

Re-stated in words, if the distance between the validation and the tuning data set is much greater than the distance between the validation data set and its simulation, then that validation is a strong validation data set. If the data set is a strong validation data set, then the following will likely hold true

\[ ||\Phi_{vi} - \Phi_{ti}|| \approx ||\Phi_{svi} - \Phi_{tsi}||. \]

This property will be used in Section 5.5.3 to predict the validation data set. To maximize the likelihood of the aforementioned occurring, one needs to change the behavior of the test vectors. In this work, the voltage command will not vary in the tuning data set where the future data set and the validation data set will have a varying voltage output command. It was possible in this work to use similar validation data as the future data set. In that case one would then select tuning data that is quite different than the validation data set. In situations where selection of the validation data set is not fixed to the future data set (e.g., not able to obtain flight data for validation of a component), one could maximize the uncertainty functions to select validation data “farthest” away from the tuning data set. The future data set could then be estimated based on work from Section 5.5.3 (i.e., the future data set becomes a new validation data set, and the tuning and validation data set become the new tuning data set).
Figure 4.2: Example of weak validation data (red x indicates farthest point), distance = 0.0775.
Figure 4.3: Example of stronger validation data (red x indicates farthest point), distance = 0.5240.
Note, because of the nature of the minimum and maximum calculation, the distance function specified is dependent on the order of the vector specified, so the notation from Eq.(4.7) will be followed throughout. Also note, that since the distance function is dependent on the functions used for construction there will be a superscript assigned to each indicating the set of data used for tuning and the type of uncertainty (either model or control), so $||\Phi_{svi} - \Phi_{ti}||^{mt}$ is based on the model uncertainty functions created from the data set $D_t$. Also note, the quantity $||\Phi_{svi} - \Phi_{ti}||^{mv}$ is tuned from the tuning data set and the validation data set.

4.1.3 Uncertainty Limits

The uncertainty limits $\Delta_{mi}(\Phi_{vi})$ and $\Delta_{ci}(\Phi_{si})$ are determined through use of a tolerance interval on the data set acquired and the following is a description of that process and motivation as to why a tolerance interval is used instead of a supremum calculation. Define a sample population of test runs characterized by $\Phi_{jik}$, where $k = 1 \ldots N$, where $k$ is a test run in the data set $j$. Calculate a supremum over each test run $k$ for each uncertainty limit

$$\Delta_{mik}(\Phi_{jik}) = ||\delta_{m}(\Phi_{jik}, t)||_{\infty} = ||\dot{x}_{i}(\Phi_{jik}, t) - \dot{x}_{mi}(\Phi_{jik}, t)||_{\infty},$$

$$\Delta_{cik}(\Phi_{jik}) = ||\delta_{c}(\Phi_{jik}, t)||_{\infty} = ||\dot{x}_{mi}(\Phi_{jik}, t) - \dot{x}_{ci}(\Phi_{jik}, t)||_{\infty},$$

where $\dot{x}_{i}$, $\dot{x}_{mi}$, and $\dot{x}_{ci}$ use the derivative calculation described in Eq.(7.1). Next, calculate an upper bound, $\Delta_{m,ci}(\Phi_{j})$ on $p_1$ fraction of the sample population of $\Delta_{m,cik}$ with confidence of $\alpha$ using a tolerance interval such that

$$P(F(\Delta_{m,ci}(\Phi_{j})) > p_1) > \alpha.$$ 

Further, assume the population includes all test vectors such that

$$||\Phi_{jin} - \Phi_{ji}|| < \epsilon,$$

where $\epsilon$ is a small value, then for the population of $\Phi_{jin}$ of such test vectors, the upper bound $\Delta_{i}$ will hold. Although this does not give the supremum over the population, as it would require a
potentially infinite number of test runs, it can meet the risk requirements that will be described in Section 5.5.2. For a discussion on the implementation used for calculation of the tolerance interval see Appendix B.

4.1.4 Total Uncertainty Estimates

In this section two other methods of uncertainty estimation will be discussed and used for future comparison to what was developed earlier. So for reference, Eq.(4.6) was

$$\delta_i(\Phi_{fi}, t) \leq L_{mi}\|\Phi_{vi} - \Phi_{ti}\| + \Delta_{mi}(\Phi_{vi}) + L_{ci}\|\Phi_{si} - \Phi_{ti}\| + \Delta_{ci}(\Phi_{si}).$$

The method in Eq.(4.6) will be referred to as the Lipschitz proof estimate. It was found in the example problems, to overestimate the uncertainty. So an alternative to the distance estimation will be used. Previously, $\|\Phi_{fi} - \Phi_{vi}\| \leq \|\Phi_{vi} - \Phi_{ti}\|$ was used, what is being proposed is that $\|\Phi_{fi} - \Phi_{vi}\| \sim \|\Phi_{si} - \Phi_{vi}\|$ for the model uncertainty and $\|\Phi_{fi} - \Phi_{si}\| \sim \|\Phi_{si} - \Phi_{vi}\|$ for the control uncertainty. With that change, the total uncertainty estimate is

$$\delta_i(\Phi_{fi}, t) \leq L_{mi}\|\Phi_{si} - \Phi_{vi}\| + \Delta_{mi}(\Phi_{vi}) + L_{ci}\|\Phi_{si} - \Phi_{vi}\| + \Delta_{ci}(\Phi_{si}).$$  (4.8)

This method will be called the Lipschitz approximate estimate.

The final total uncertainty estimate that will be used is one that is primarily based on the regression analysis of the uncertainty. That is,

$$\delta_i(\Phi_{fi}, t) = \delta_{mi}(\Phi_{fi}, t) + \delta_{ci}(\Phi_{fi}, t).$$  (4.9)

This estimate will be called the uncertainty function estimate. The last uncertainty estimate is here to demonstrate, that when the regression equation is used to estimate the total uncertainty, there is a higher requirement on its accuracy to predict new data. Further, the other two uncertainty estimates only rely on having a close estimate of the partial derivative of the uncertainty equation and not the actual values.
4.2 Simulation Setup

Two similar models are used throughout this investigation. To generate new data for prediction, a simulation which will be called an integrator in the loop simulation, or closed loop simulation, is used. This simulation makes use of an integrator to update the state information (see Fig. 4.4) used with the differential equations. The second type of simulation will be called an integrator out of the loop simulation, or open loop model. The state information in this simulation is obtained from experimental data and the derivatives drive an integrator to generate new state data, without the new state data affecting the derivative calculation (see Fig. 4.5). This last model was created to allow for accurate analysis of the derivative calculation by making use of the state information directly from the experimental data. Any differences between the model and the experimental data will be limited to the differential equations and not state changes that can occur over time.

![Figure 4.4: Integrator in the loop simulation model (closed loop model)](image)

Model validation in practice is often performed through use of models that maintain integrators in the loop simulations and evaluate the outputs. The analysis of the uncertainty here requires an
integrator out of the loop simulation. If one used an integrator in the loop simulation for the analysis above, the following values for total uncertainty would be found

\[
\delta_{mi} \equiv \dot{x}_i(\Phi_{ji}) - \dot{x}_{mi}(\Phi_{ji}),
\]

\[
\delta_{ci} \equiv \dot{x}_{mi}(\Phi_{ki}) - \dot{x}_{ci}(\Phi_{ki}),
\]

\[
\delta_i = \delta_{mi} + \delta_{ci} + \dot{x}_{ci}(\Phi_{ki}) - \dot{x}_{ic}(\Phi_{ji}),
\]

(4.10)

with the last two items in Eq. (4.10) representing a source of error. In the above equations $\Phi_{ji}$ represents the data from the experimental testing, and $\Phi_{ki}$ represents data from the simulation. Example problem 2 (see Section 6.2) will demonstrate this issue.
CHAPTER V

DC-DC CIRCUIT MODEL APPLICATION

5.1 DC-DC Boost Circuit Model and Control Development

In this chapter, the application of using the dc-dc boost circuit model with the generalized approach described previously will be discussed. This chapter will describe the model development, the simulations used, the control development, and discuss the validation process.

5.1.1 Model Development

The dc-dc circuit model used here is based on an average value formulation from [64]. As described in the reference, the n-MOSFET switch and diode are replaced with voltage and current sources as depicted in Fig. 5.1 to construct the average value model. It assumes only series resistance losses in the components of the system [64]. Sense resistors are added that allow for measurement of key currents within the circuit. For this system, it is assumed that the output current \( I_o \), derivative of the output current \( I_o \), and input voltage \( V_i \) can be measured for control. These signals are considered boundary conditions to the system. With these assumptions, the differential
The above equation represents a first order representation of the diode drop. A second order representation is the following

\[
\frac{dI_l}{dt} = \frac{V_i - I_l r - I_o r_{sns2} + V_F (d_T - 1) + V_o (d_T - 1)}{L},
\]

(5.2)

\[
r = r_l + r_{sns1} + d_T r_{ds} - R_F (d_T - 1),
\]

(5.3)

\[
V_o = V_c - I_o r_{sns2} - r_c (I_o + I_l (d_T - 1)).
\]

(5.4)

The derivations for the second order diode drop equation can be found in Appendix D. The need for the second order diode model is discussed in Section 7.3.1.
5.2 Simulation Setup

All simulations were performed using Mathwork’s Simulink environment using version 2014a. The solver used was the “ode23tb”. The maximum step size was set to 1/4 of the inverse of the control frequency of 14 kHz. A graphic of the open loop simulation can be found in Fig. 5.2. The “Backstepping Controller” block will be discussed in detail in Section 5.3. The “dc-dc circuit Plant” uses the differential Eqs.(5.1, 5.3, 5.5, 5.6). The block “ControlStateCalculations” performs the calculation

\[ \dot{x}_{c1} = f_1 + g_1 x_2, \]
\[ \dot{x}_{c2} = f_2 + g_2 u, \]

where \( \dot{x}_{c1} \) is the voltage output control differential, \( \dot{x}_{c2} \) is the inductor current control differential, \( x_2 \) is the inductor current, and \( u \) is the duty cycle, \( d_T \). The functions \( f_1, g_1, f_2, \) and \( g_2 \) will be discussed in the controller development section. All the low pass filters (LPF) are 10 kHz Butterworth two-pole filters. These filters are close approximations of the filters found on the data acquisition cards.

The closed loop model is depicted in Fig 5.3. The differences between these models primarily involve the routing of the outputs “Il_data” and “Vo_data”. This difference follows the discussion made in Section 4.2. Further, as will be discussed in more detail in the example problems there is a zero-order hold block in the closed loop model necessary to allow for matching between the open loop and closed loop model. The need for the block “Duty_Cycle_Mod” and “Transport Delay” will be discussed in Section 5.5.3.

One issue or difficulty faced with setting up a simulation in this manner especially with hardware data is collection of state information that may not be directly measurable as in this case. Both the inductor current and the voltage across the capacitance of the capacitor cannot be directly measured. In the case of the inductor current, the estimate is rather good, due to the fact one can measure the
voltage across the sense resistor and estimate the current. The calculation of that uncertainty will be done in Section 7.3.2. The more difficult estimation is the voltage across just the capacitance. Fortunately, in this case the state equations can be rewritten so as to not depend on the voltage across the capacitance, but only the output voltage. The state equation in terms of $V_o$ is written as
\[
\dot{V}_o = \dot{V}_c - I_o r_{sns2} - r_c \left( \dot{I}_o + \dot{I}_l \left( d_T - 1 \right) + I_l \dot{d_T} \right)
\]  
(5.6)
where $\dot{V}_c$ is Eq.(5.1) and $\dot{I}_l$ is Eq.(5.5). Other options to solving the problem include use of an observer [36].

5.3 Control Development

One goal of this work is to develop control algorithms that ensure good power quality. Three methods of control will be discussed and two of the methods, backstepping control and augmented backstepping control, will be implemented. The third method is nonlinear damping which is discussed in Appendix C.3.
5.3.1 Backstepping Control

Backstepping control was chosen because of the classical means of controlling dc-dc boost converters that rely on development of an outer loop voltage controller that commands an inner loop current controller. This decomposition of the control fits well with the backstepping control approach.

The first problem encountered with using backstepping control is that the differential equations describing the dc-dc boost circuit (see Eqs. (5.1-5.5)) are not in strict feedback form because of the $d_T$ term in Eq. (5.1). To overcome this issue, a steady state approximation for $d_T$ is taken such that

$$d_T = \frac{V_o - \eta V_i}{V_o}$$

(5.7)

where $\eta$ is the average efficiency of the dc-dc boost circuit. The differential equation used for control is

$$\dot{x}_1 = -\frac{I_o}{C} + \left(\frac{V_i \eta}{C x_1}\right) x_2.$$
Equation 5.1 was used in the development of the control instead of Eq. (5.6) because of the $\dot{d}_T$ term.

Applying backstepping control to the problem results in the following \cite{45}

\[ u = 1/g_2(-f_2 + \dot{\alpha}_c + k_2e_2 - \frac{g_1}{\beta_c^2}e_1), \quad (5.8) \]

where

\[ e_1 = x_1 - x_{1r}, \]
\[ e_2 = x_2 - \alpha_c, \]
\[ \alpha_c = 1/g_1(-f_1 - k_1e_1 + \dot{x}_{1r}), \quad (5.9) \]
\[ \dot{\alpha}_c = \frac{\partial \alpha_c}{\partial t} + \frac{\partial \alpha_c}{\partial x_1}(f_1 + g_1x_2), \quad (5.10) \]
\[ f_1 = \frac{I_o}{C}, \]
\[ g_1 = \frac{V_i\eta}{C_1}, \]
\[ f_2 = -\frac{V_F + x_1 - V_i + x_2(R_F + r_t + r_{sns1}) + I_o r_{sns2} + D_F x_2^2}{L}, \]
\[ g_2 = \frac{V_F + x_1 + x_2(R_F - r_{ds}) + D_F x_2^2}{L}, \]

where $x_{1r}$ is the reference command and can vary, $x_1$ is the output voltage $V_o$, and $x_2$ is the inductor current $I_l$. For simplicity it is assumed that the input voltage is constant. With a time varying input voltage, the backstepping control law will result in a divide by zero since $\dot{\alpha}_c$ would be a function of $\frac{1}{V_i}$ and $\dot{V}_i$ will likely be zero or very small at some point. Further, the gains $k_1$, $k_2$, and $\beta_c$ are
greater than zero. This control results in the error dynamics

\[
\dot{e}_1 = \dot{x}_1 - \dot{x}_{1r}
\]

\[
= f_1 + g_1 x_2 + \delta_1 - \dot{x}_{1r},
\]

\[
x_2 = e_2 + \alpha_c,
\]

\[
\dot{e}_1 = f_1 + g_1 (e_2 + \alpha_c) + \delta_1 - \dot{x}_{1r},
\]

\[
= f_1 + g_1 (e_2 + 1/g_1(-f_1 - k_1 e_1 + \dot{x}_{1r})) + \delta_1 - \dot{x}_{1r},
\]

\[
= -k_1 e_1 + g_1 e_2 + \delta_1,
\]

and

\[
\dot{e}_2 = \dot{x}_2 - \dot{\alpha}_c,
\]

\[
= f_2 + g_2 u + \delta_2 - \dot{\alpha}_c,
\]

\[
= f_2 + g_2 (1/g_2(-f_2 + \dot{\alpha}_c - k_2 e_2 - g_1/\beta_c e_1)) + \delta_2 - \dot{\alpha}_c
\]

\[
= -k_2 e_2 - g_1/\beta_c e_1 + \delta_2.
\]

The term \(\dot{\alpha}_c\) found above is an estimate only, as the value of \(\alpha_c\) includes a function of \(\delta_1\), which is unavailable. Likewise \(\dot{e}_2\) has similar behavior so only the estimate \(\hat{\dot{e}}_2\) is used for control. Mathematically \(\dot{\alpha}_c\) is

\[
\dot{\alpha}_c = \frac{\partial \alpha_c}{\partial t} + \frac{\partial \alpha_c}{\partial x_1} \dot{x}_1
\]

\[
= \frac{\partial \alpha_c}{\partial t} + \frac{\partial \alpha_c}{\partial x_1} (f_1 + g_1 x_2 + \delta_1).
\]

For Lyapunov stability analysis, the error dynamic for the second state \(\dot{x}_2\) is

\[
\dot{e}_2 = \dot{x}_2 - \dot{\alpha}_c
\]

\[
= -k_2 e_2 - g_1 e_1 + \delta_2 + \dot{\alpha}_c - \dot{\alpha}_c.
\]
Once the bounds on the uncertainties are found, one can use those limits in a Lyapunov stability analysis to ascertain the stability of the system at a regulation point and depending on the results change the control law to meet stability requirements. For example, if maximum constant uncertainties are found such that
\[ \Delta_1 \geq |\delta_1|, \Delta_2 \geq |\delta_2|, \text{ and } W_3 \geq \left| \frac{\partial \alpha}{\partial x_1} \right| \]
are identified, then using a Lyapunov candidate function one finds the following

\[
V_1 = \frac{1}{2} e_1^2 \quad (5.11)
\]

\[
\dot{V}_1 = e_1 \dot{e}_1 = e_1(-k_1e_1 + g_1e_2 + \delta_1)
\]

\[
\leq -k_1 e_1^2 + g_1 e_1 e_2 + \delta_1 e_1,
\]

\[
V = \frac{1}{2} e_1^2 + \frac{1}{2} (\beta_c e_2)^2 = V_1 + \frac{1}{2} (\beta_c e_2)^2 \quad (5.12)
\]

\[
\dot{V} \leq -k_1 e_1^2 + g_1 e_1 e_2 + \Delta_1 |e_1| + \beta_c^2 e_2 (\dot{e}_2)
\]

\[
\leq -k_1 e_1^2 + \Delta_1 |e_1| + g_1 e_1 e_2
\]

\[
\quad + \beta_c^2 e_2 (-k_2 e_2 - \frac{g_1}{\beta_c^2} e_1 + \delta_2 + \frac{\partial \alpha}{\partial x_1} \delta_1)
\]

\[
\leq -k_1 e_1^2 - k_2 \beta_c^2 e_2^2 + \Delta_1 |e_1| + \beta_c^2 e_2 |e_2| - \beta_c^2 \frac{\partial \alpha}{\partial x_1} \delta_1 e_2
\]

\[
\leq -k_1 e_1^2 - k_2 \beta_c^2 e_2^2 + \Delta_1 |e_1| + \beta_c^2 W_3 |e_2| + \beta_c^2 W_3 A_1 |e_2| \quad (5.13)
\]

\[
\leq -(k_1 - 1) e_1^2 - (k_2 - k_{2o}) \beta_c^2 e_2^2 - e_1^2 + \Delta_1 |e_1|
\]

\[
- k_2 \beta_c^2 e_2^2 + \beta_c^2 (\Delta_2 + W_3 A_1) |e_2|
\]

\[
\leq -(k_1 - 1) e_1^2 - (k_2 - k_{2o}) \beta_c^2 e_2^2 + \frac{\Delta_1^2}{4}
\]

\[
+ \beta_c^2 (\Delta_2 + W_3 A_1)^2 \frac{e_2^2}{4k_{2o}}.
\]

The boundary on the Lyapunov candidate function can be written as

\[
V = \frac{1}{2} e_1^2 + \frac{1}{2} (\beta_c e_2)^2,
\]

\[
\lim_{t \to \infty} |V(t)| \leq \frac{W_d}{k_m}. \quad (5.14)
\]
From the equation above, one can then show that $e_1$ is bounded by

$$\lim_{t \to \infty} |e_1(t)| \leq \sqrt{\frac{2W_d}{k_m}},$$

(5.15)

where

$$k_m = \min(k_1 - 1, k_2 - k_{2o}),$$

$$W_d = \frac{\Delta_1^2}{4} + \frac{\beta_c^2(\Delta_2 + W_3\Delta_1)^2}{4k_{2o}}.$$  

(5.16)

Thus, depending on whether the calculated bound meets the requirements for the system, one could either apply nonlinear damping or incorporate the control uncertainty into the original control to reduce the bounds on $e_1$.

### 5.3.2 Beta Modification

Early on in the development it was found that by adding what will be called beta modification one can modify the invariant set to meet the required needs. The term $\beta_c$ added into Eq.(5.16) is the beta modification term. This term is also part of the control law (see Eq.(5.8)). The beta term can best be viewed as a weighting function on the importance in maintaining the inductor current state versus the output voltage state. As the term $\beta_c$ increases, the invariant set in respect to current will reduce as seen in Figs. 5.4 and 5.5. In Fig. 5.4, the limit on current is beyond the one ampere displayed, while in Fig. 5.5, the limit on the current is 0.85 A. The maximum voltage error is slightly larger with the higher value of beta. Fortunately, the invariant set is much smaller. This beta modification allows the engineer some flexibility in selecting gains for molding the invariant set to the desired shape.

### 5.3.3 Augmented Backstepping Control

If one chooses, the identified control uncertainty, $\delta_{ci}$, can be used within the backstepping framework, assuming that the augmented system can be represented in strict feedback form. For example,
if $\delta_{c1}$ is a function of $u$ or a nonlinear function of $x_2$ then augmentation is not possible. The control development follows the same development as backstepping control with the modified nonlinear differential equations. This augmentation will be shown to reduce the control uncertainty and thus total uncertainty.

### 5.4 Domains and Data Sets

For all the testing in the example problems and with the experiment, the following data sets were created.
• $D_t$ is the tuning data set, it is a set of 32 test vectors with constant output voltage commands ranging from 4.5 V to 9.5 V, constant source voltages ranging from 3 V to 4 V, and a load transition from one load resistor to two load resistors or vice versa at a rate of about ± 40 A/s at a time of 0.1 s. This test matrix was created using DesignExpert [47] using a D optimal design. The number of test runs was calculated from [9] with $\alpha = 0.05$ and $\beta = 0.01$. The uncertainty model considered was a second order three term model. This specification along with six center points and three replications resulted in 32 test vectors. Plots of a typical test run can be found in Figs. 5.6, 5.7, 5.8, and 5.9. See Appendix A for the test vectors.
Figure 5.6: Plot of the output voltage ($V_o$) of a typical test vector in $D_i$.

- $D_{sv}$ is the prediction for the model validation data set and thus is a simulation generated data set. It is a 76 test vector data set where the voltage command $V_{oref}$, starts from a value between 4.5 V to 9.5 V and is switched to 7 V at a time of 0.05 s with a low pass filter with a bandwidth of 10 Hz. The range of input source voltage, $V_i$, is 3 V to 4 V. The load transition is similar to what was used for the tuning data set. This data set is a randomly chosen data set using a Halton sequence [48]. The number of points was chosen to meet the risk requirements. It was designed as a strong validation data set by adding the change in $V_{oref}$. See Appendix A for a list of test vectors.
• $D_v$ is the model validation data set and uses the same boundary conditions as $D_{sv}$ but is an experimental data set. Plots of a typical test run can be found in Figs. 5.10, 5.11, 5.12, and 5.13.

• $D_s$ is a simulated data set based on using the augmented backstepping control. It is the prediction data set for the future data set. It uses the same boundary conditions as $D_v$ (i.e., uses the actual $V_i$ from that data set).

• $D_f$ is the future data set to be taken, it uses the same test vectors that were used to generate the validation data. This data uses the augmented backstepping control. This data set is what is being predicted by using the validation process.
5.5 Process

The validation process used here is based upon the documentation specified in MIL-STD 3022 [11]. It is composed of five primary steps with four secondary steps under the verification and validation primary step (see Fig. 5.14). The work here combines the modeling assumption step with the requirements step that is separate under MIL-STD 3022. The process presented here leaves out some of the non-technical aspects from MIL-STD 3022 like executive summary and list of key participants. This section will discuss these documentation steps and how they apply to the model and control validation of the dc-dc boost circuit.
5.5.1 Problem Statement

This section will follow the outline that is specified in MIL-STD 3022. Specifically, it should focus on the intended use of the model, an overview of the model, how the model will be used in the larger program, and the scope of the verification and validation effort. As mentioned in MIL-STD 3022 it should be noted as to the issues that might occur if the model reports the wrong information (i.e., type II error). For the dc-dc circuit the following would suffice for the problem statement.

The purpose of the dc-dc boost circuit model is for control algorithm development and validation of the control for a non-safety critical application. This model along with the control will be used to
assess the quality of the output voltage signal (i.e., the amount of disturbance from the commanded voltage). Estimation of the model and the control algorithm uncertainties will be used in Lyapunov stability analysis to predict performance of the control.

### 5.5.2 Requirements

This section combines the modeling assumptions and requirements from MIL-STD 3022 into a single requirements section. The modeling assumptions from MIL-STD 3022 are considered modeling requirements and the requirements section from MIL-STD 3022 addresses risk requirements and accuracy requirements. By setting the requirements up-front, an idea of when the modeling
effort can stop is accomplished and will inform the engineer to when “good” is “good enough”, without adding unnecessary cost.

**Risk Requirements**

To define risk requirements for model validation, it is paramount that the intended use of the model, a tolerable risk level, and model accuracy be understood. Unfortunately, this dissertation cannot specify what those requirements need to be for every modeling effort. This dissertation will outline some general thoughts on items to be considered and will present an explanation for the requirements for this problem.
The first requirements to be obtained or developed are those associated with risk and sampling in the validation exercise. For this work, the risk allowed for Type I error is 5% (i.e., $\alpha = 0.05$) and the allowable risk for Type II error is 1% (i.e., $\beta = 0.01$). For sampling purposes, the acceptable quality level (AQL) will be set to 0.05 (i.e., $p_1 = 0.05$). The rejectable quality level (RQL) will be set to 0.2 (i.e., $p_2 = 0.2$). In words, “If the model tested can predict more than 95% of the future data ($1 - AQL$), we will accept that model as “good” 95% ($1 - \alpha$) of the time”. “If the model predicts less than 80% of the data not taken ($1 - RQL$) the model will be rejected as “bad” 99% ($1 - \beta$) of the time.”.

Figure 5.12: Plot of the load current ($I_o$) of a typical test vector in $D_v$. 
Figure 5.13: Plot of the source voltage ($V_i$) of a typical test vector in $D_v$. Commanded input voltage was 3.0694 V.

In Fig. 5.15 a number of acceptance plans are depicted indicating different qualities that the choice of the acceptance plan can exhibit. For example, the optimal plan used here ($n = 76, c = 7$) lies almost on top of the appropriate alpha level for acceptable quality level and beta for the rejectable quality level (i.e., the two red circles in Fig. 5.15). If only 10 test points are available for validation, then it would be up to the user to make the trade-off between allowable risk between Type I and Type II errors. Obviously as depicted if more than the prescribed number of points are used then the acceptance plan becomes better as shown with the trace (e.g., with $c = 70, n = 760$). An optimal acceptance plan is one that minimizes the risk involved in making a Type I or Type II error as described in [46].
How one determines the allowable risk is through understanding the problem of interest. The levels stated here are relevant to problems with low safety concerns or low economic impact (e.g., if the model gives the wrong answer there is little or no loss in safety). If safety or high economic impact is of concern then the risk levels would need to be made more stringent (e.g., $\alpha \sim 1\%$). Also note, if a set budget is given, then based on the number of tests that can be allocated for validation based on that budget, a rough estimate to the risk levels that can be achieved could be stated (e.g., if 10 tests were permitted and one chose $c = 0$, then per Fig. 5.15 $\alpha = 0.4$ with AQL = 0.05 and $\beta = 0.12$ with RQL = 0.2).
Accuracy Requirements

The next set of requirements is selection of the system response quantities (SRQs) and specification of limits on the SRQs. This requires a good understanding of the model’s intended use. Accuracy can either be set through a requirements decomposition of system level requirements as done here or through subject matter experts. The system response quantity of interest for the dc-dc boost circuit model validation is the output voltage. For this problem, there is a desire to control the boost circuit to within $\pm 200$ mV of the desired voltage level. To achieve those results, requirements on the state uncertainty in the control and model were determined to be 12 V/s and 150 A/s each from Lyapunov stability analysis (with $\beta_c = 1$). It was decided the requirements of uncertainty for
control will be smaller than the requirements of uncertainty for the model. The control uncertainty is set at 3 V/s and 50 A/s and thus allow for 9 V/s and 100 A/s for model uncertainty. Background for developing this criteria can be found in [77]. Note, the goal in [77] was to track to 50 mV and only accounted for the control uncertainty. The modeling uncertainty is expected to be larger than the control uncertainty.

It will be shown in Chapter VII that the uncertainty in the measurement system is 7.7 V/s which gives little to no opportunity to demonstrate the model uncertainty is less than 9 V/s. So the requirements will need to be changed. In fact, early testing indicated the modeling uncertainty is estimated to be 90 V/s. So rather than rely on using Lyapunov stability analysis to demonstrate the tracking requirement, the goal is to ensure the system is stable within a larger region (i.e., 7 ± 2.5 V). To meet those goals, Δ1 should be less than 150 V/s and Δ2 should be less than 300 A/s. To decompose the problem to a model uncertainty limit and control uncertainty limit, the requirements for each will be Δm1 < 100 V/s, Δm2 < 150 A/s, Δc1 < 50 V/s, and Δc2 < 150 A/s. Figure 5.16 demonstrates the invariant set for those requirements with βc = 15. Notice, the voltage stays less than 8 V. The low voltage is not a concern as the physics prevent the voltage from dropping below the supply voltage.

The final testing will demonstrate that the region of attraction is much smaller than what is predicted by Lyapunov stability analysis and that the 200 mV tracking is met. In fact the baseline controller has an invariant set of less than ± 40 mV as seen in Figs. 5.17. The red ’+’ indicates the first point in the series, the blue line is indicative of the movement within the invariant set, and the green ’o’ are indication of the end state.
Figure 5.16: Demonstration of the invariant set for the Lyapunov stability analysis (green is the invariant set).

**Modeling Requirements**

The modeling requirements specify the future operating conditions and any other modeling assumptions. Typically it is necessary to limit the scope of validation (due to cost of testing), and intended use of the model. For example, if the item under consideration is only ever used at room temperature, then it is not necessary to perform testing at $-40^\circ$ C, let alone including temperature within the model. Understanding and stating the limitations on the operating conditions up front is critical. It will also allow future users to understand the limitations and risks in using this model.

For the work here, the inputs that will be changed to induce a change in the output are the reference
voltage level (i.e., the commanded output voltage), the input voltage level, and the load current direction (i.e., either a switch from one resistor to two resistors or two resistors to one resistor).

The model prediction space will use a voltage output command that starts from a range of 4.5 V to 9.5 V and then switches to 7 V with a filter time constant of 10 Hz (i.e., data set $D_f$). The prediction space will include use of the new control law.

The model used here is an average value model developed from [64] that is accurate in continuous conduction mode. If the inductor current reaches zero the model accuracy will be questionable. Prevention of the inductor current tending towards zero is the reason for the addition of the 10 Hz filter on the command in the validation data. Finally the frequency of interest in the analysis is 5

Figure 5.17: Demonstration of the invariant set for the baseline system within the domain $D_v$. 
kHz which is much less than the 100 kHz switching frequency and as such the average value model is likely to be a good representation of the system.

5.5.3 Model Verification and Validation

Data Uncertainty Quantification

The data uncertainty quantification should be used to determine and document the parameter uncertainty, measurement uncertainty, and input uncertainty. For this problem, these items are documented in Chapter [VII]. These quantities are important to evaluate whether the requirements will likely be met. They can help identify the need for better measurements of parts or sensors to be used on the system.

Model Qualification

Model qualification involves ensuring that the model has been designed to the modeling requirements. It will answer the question, “Will all the modeling requirements likely be met?” This is not an evaluation of the implementation of the model as that will be covered in model verification, but a requirements verification check. This step was not formally done for the dc-dc boost circuit model. During design of the model items like ensuring the model would allow for varying of the input voltage, the model made use of the continuous conduction mode modeling practices as described in [64], and that the resistor could be switched from 100 to 200 Ω were performed. For larger modeling programs where there may be hundreds of model requirements, there may be need of a formal requirements tool and then linking of requirements to sections of the model that fulfill those requirements. In [11] and [56] this step is referred to conceptual model validation.
Model Verification

Model verification is the determination that the implementation is done correctly. There are a number of means to perform model verification, two references to review are [18], [78]. The primary means of verification in this project was comparison of the dc-dc boost circuit model with other modeling tools and means of modeling. Those comparisons included use of National Instruments MultiSim tool; two different a-causal modeling tools, Simscape and ASMG; and an analytical model of the dc-dc boost circuit. Through these comparisons, the model was deemed verified.

Model Validation

The process of model validation is broken into the four steps of screening and exploration, system and/or parameter identification, prediction, and hypothesis testing.

Screening and Exploration  Within the screening and exploration process, three topics need to be considered. The first topic, exploration, involves understanding general behavior of the system. The second topic that screening and exploration addresses is the determination of the relevant inputs to the experiment and their effect on the output. The third topic is determining whether additional system identification and parameter tuning is necessary (i.e., the model is good enough as it is without additional work).

The findings of the screening and exploration of the dc-dc boost circuit were the following. First, as will be discussed in Chapter VII [VII], the update rate of the controller was reduced from a desired 20 kHz to 14 kHz to ensure the controller met its loop time. Further, it was found that the controller was only able to drive one pulse width modulated (PWM) output at a time, reducing the ability to perform certain test (i.e., not able to command a chirp to both the duty cycle and the output n-MOSFET to perform a more rigorous tuning experiment).
The screening work of the relevant inputs and parameters found that all three boundary conditions (i.e., the output voltage command, the source voltage, and the load transition direction) were relevant and through some prior work [79] found that the model was insensitive to the series resistance of the n-MOSFET parameter, so it will not be used in parameter optimization.

The final analysis examines whether there is a need for system identification or tuning. The results of the preliminary testing of dc-dc circuit model indicates model tuning is necessary. Further discussion will be presented in Chapter [VIII].

**System and/or Parameter Identification** System and/or parameter identification is heavily dependent on a subject matter expert understanding the physics displayed in the data and the physics captured in the model. It is typically up to that expert to determine whether additional parameter identification is necessary, additional physics should be captured in the model, or if there is error in the data acquisition / testing when the accuracy requirements are not met. The required accuracy is important to determine when this step is finished. If requirements are not met, one option is to review with the stakeholders whether the requirements can be relaxed so as to proceed. Throughout this step, tolerance intervals and predictions should be consistently re-calculated with changes and improvements.

The system and/or parameter identification step can be highly iterative between parameter tuning, increased fidelity, and fixing data collection problems. Several steps of parameter identification could be made for every change in the model or data collection that is done.

Before parameter identification, a number of tests were performed to understand whether additional physics were required by the model. Three items were found and added to the model. The first change was the use of a higher order model of the diode (see Section [7.3]). The second item was the need for a scalar and offset to the duty cycle command. This effect was added to better model
the non-ideal behavior of the n-MOSFET when commanded with a PWM signal. See the work in [80] for further discussion. The last item found was the delay in the control to commanding a PWM signal to the n-MOSFET. Through use of an analog output and an additional analog input on the DAQ system this delay was confirmed. It was originally noticed when using a chirp signal as the duty cycle command. There still remains some un-modeled dynamics, one of note that was found in [80] was an apparent difference in duty cycle behavior between use of two resistors or one resistor possibly stemming from the second n-MOSFET. Further investigation is necessary before adding physics to the model.

The optimization of parameters step is focused on identification of the parameters of the experimental plant in light of the lack of complete knowledge of the physics that should be modeled. Model bias is thus an issue [50]. Although not solved here, one of the means of reducing the effect of model bias on the parameter selection is through limitation of the parameter variation based on knowledge of the system (see Table 7.3 for the limits used in this project). This method will be highlighted in Chapter VIII. Though, this limitation may not capture all the installed effects versus stand alone testing [17].

For the dc-dc boost circuit model, the two techniques for parameter optimization that were analyzed include fmincon which is a nonlinear constrained optimization [81] and a genetic algorithm [82], [81]. There are other means of optimization, but these methods were readily available within Matlab. Other comparisons were also done based on the optimization cost function.

In this work, the two cost functions used for the parameter optimization were

\[ J_x = \sum_{j=1}^{M} \frac{||e_{V_o}(j)||}{\sqrt{N}} + \sum_{k_s}^{M} \frac{||e_{I_l}(j)||}{\sqrt{N}} k_s \text{ for each test run } j, \]  
(5.17)

and

\[ J_{\dot{x}} = \sum_{j=1}^{M} \frac{||\dot{e}_{V_o}(j)||}{\sqrt{N}} + \sum_{k_s}^{M} \frac{||\dot{e}_{I_l}(j)||}{\sqrt{N}} k_s \text{ for each test run } j, \]  
(5.18)
where

\[
e_{V_o}(j) = V_{om}(j) - V_o(j),
\]

\[
e_{\dot{V}_o}(j) = \frac{\lambda p}{p + \lambda} V_{om}(j) - \frac{\lambda p}{p + \lambda} V_o(j),
\]

\[
e_{I_l}(j) = I_{lm}(j) - I_l(j),
\]

\[
e_{\dot{I}_l}(j) = \frac{\lambda p}{p + \lambda} I_{lm}(j) - \frac{\lambda p}{p + \lambda} I_l(j),
\]

where the \( p \) operator is from state variable filtering, \( \lambda \) is a parameter for filtering with a cutoff frequency of 5 kHz, \( N \) is the number of data points for a given test, \( M \) is the number of tests in a data set, \( V_{om}(j) \) is the array of model voltage outputs for test \( j \), \( V_o(j) \) is an array of experimental voltage outputs for a test \( j \), \( I_{lm}(j) \) is the array of model inductor currents for a test \( j \), \( I_l(j) \) is an array of experimental inductor currents for a test \( j \), and \( k_s \) is a scaling term to either add or reduce weight of the inductor current terms in the optimization.

In the example problems, \textit{fmincon} will be used with the cost function \( J_x \) (i.e., Eq. (5.17)). The other methods will be used in Section 8.1.

In addition to the parameter optimization, in this step of the validation process, determination of the control and model uncertainty will be made with the tuning data set. The development of models for \( \delta_{m1} \) and \( \delta_{m2} \), the analysis of those models for use in predicting the uncertainty of future data, and calculation of the uncertainty limits is done here (i.e., calculation of \( \Delta_{m1}(\Phi_{t1}) \) and \( \Delta_{m2}(\Phi_{t2}) \)). In this stage, data set \( D_t \) is used extensively.

If the model developed in the previous stages is in strict feedback form, then the model can be used directly for control and there will be no control uncertainty as will be seen in the first two example problems. However, if the model cannot be put into strict feedback form, then the potential of control uncertainty exists, and can be used in an augmented controller. The goal of this stage would be to develop the controller uncertainties \( \delta_{c1} \) and \( \delta_{c2} \) and determine whether the estimated
uncertainty is acceptable to continue or if there is a need for development of an augmented control. See Section 5.3.3 for implementation of the augmented backstepping controller.

Finally, the control uncertainty will be re-calculated based on use of the augmented controller. This will be done with data from the model validation data set (i.e., $D_v$). It determines uncertainty estimates $\delta_{c1}$ and $\delta_{c2}$ of the augmented control. These uncertainties are used to calculate the Lipschitz constant and distance between the domain $D_v$ and the tuning domain $D_t$, $||\Phi_{vi} - \Phi_{ti}||^{cv}$. In this stage, measurements of the limits on the uncertainty estimates will also be calculated (i.e., $\Delta_{c1}$ and $\Delta_{c2}$ will be calculated).

The use of the data set $D_v$ here could have been replaced with $D_{sv}$ if the validation data set had not been available. Because the augmented controller had been developed with the tuning data set, use of that data set for control uncertainty estimation is not recommended.

Prediction In the prediction stage, the uncertainty estimates are used to predict the data that will be used for validation. Predictions of the model uncertainty and the control uncertainty will be made.

1. **Model Prediction** In this step, the closed loop model (i.e., integrator in the loop model) is used to generate a prediction of the validation data which is the data set $D_{sv}$. This data is used to estimate the distance between the tuning data set $D_t$ and $D_v$, (i.e., calculate $||\Phi_{svi} - \Phi_{ti}||^{mt}$). A prediction for the uncertainty of the model validation is performed

$$
\delta_{mi}(\Phi_{vi}, t) \leq L_{mi}^{t} ||\Phi_{svi} - \Phi_{ti}||^{mt} + \Delta_{mi}(\Phi_{ti})
$$

(5.19)

where $||\Phi_{svi} - \Phi_{ti}||^{mt} \approx ||\Phi_{vi} - \Phi_{ti}||^{mt}$.

2. **Control Algorithm Prediction** The quantity $L_{ci}^{v} ||\Phi_{vi} - \Phi_{ti}||^{cv}$ and $\Delta_{ci}(\Phi_{vi})$ will be calculated using the data set $D_v$ to estimate the control uncertainty limit of the data in $D_s$. The
The following equation will be used

\[ \delta_{ci}(\Phi_{si}, t) \leq L_{ci}^{v} ||\Phi_{vi} - \Phi_{ti}||^{cv} + \Delta_{ci}(\Phi_{vi}) \tag{5.20} \]

where \[ ||\Phi_{vi} - \Phi_{ti}||^{cv} \geq ||\Phi_{si} - \Phi_{vi}||^{cv} \].

**Hypothesis Testing**  The hypothesis testing used here is based on acceptance sampling and is a two-step process that includes determination of a number of samples that should be tested (along with an associated number of allowed failed tests) and a test that can be executed that would indicate pass / fail for the model. In this work the number of samples has been determined from the requirements of \( \alpha, \beta, \text{AQL}, \) and \( \text{RQL} \) already specified. The number of tests to be run is 76 with a maximum number of test fails of less than or equal to seven. The pass / fail criteria can either be based on the model meeting the accuracy requirements for a given run or the model meeting the prediction estimates for a run. These predictions may or may not be better than the original requirements. Because not all the accuracy requirements will likely be met (e.g., the inductor current control uncertainty limit), a bad part will be considered one that does not meet the prediction estimates as opposed to the accuracy requirements. Obviously, one could compare the values of the prediction estimates against the requirements to understand the likelihood of meeting the requirements and in the example presented here, the likelihood of meeting the control uncertainty requirements is very low.

1. **Model Hypothesis Test** In this stage, the model uncertainty prediction limits (see Eq. (5.19)) along with the model are tested. If the null hypothesis test is not rejected (i.e., a proportion less than AQL of the data space is predicted by the model and uncertainty limits, \( H_o : p < p_1 \)), then the model along with the model uncertainty limits are good estimates of the model uncertainty in the validation data set and likely other similar data sets. Such a hypothesis test can help with the prediction of future model performance (i.e., if the model and uncertainty
are confirmed with the validation data set, then it is likely they could be confirmed in future
data sets assuming the Lipschitz assumption is met). If further testing will be performed, then
updated estimates of \( \delta_{m1} \) and \( \delta_{m2} \) can be made. The final prediction stage below uses the
values of \( ||\Phi_{vi} - \Phi_{ti}||^{mv} \) and \( \Delta_{mi} \) calculated from this stage. The data set used in this step is
\( D_v \).

2. **Control Hypothesis Test** During this stage, the data set, \( D_s \), is created from the closed loop
simulation with the augmented controller. Also in this stage, the control uncertainty predic-
tion limits (see Eq.(5.20)) along with the model are tested. A comparison is done between
the predicted control uncertainty as calculated in the “Control Algorithm Prediction” and the
measured uncertainty from this data set. Based on the uncertainty estimates, an evaluation
of those uncertainties can be made against the boundary conditions needed for the problem.
This boundary evaluation will be done here to evaluate the likelihood the boundary conditions
are met.

5.5.4 **Verification and Validation Issues and Recommendations**

During this stage, a report of the results found along with the likelihood of meeting the risk,
accuracy, and modeling requirements should be written. During this step, recommendation for
future testing if needed should be made. Finally, if the work was done to predict performance using
the model, then the next section’s final prediction would be included.

5.6 **Final Prediction and Test**

This step is considered to be outside of the validation process but an important step to take after
understanding the validity of the uncertainty limits of the model and control. Because the future
data is similar in boundary conditions as the validation data set, the prediction used for the final
assessment is the data from $D_s$ and likely would be for most problems. Thus with the use of the augmented control, additional experimental testing will be done and analyzed.
CHAPTER VI

EXAMPLE PROBLEMS

In this chapter several example problems will be presented and used to demonstrate the processes developed. These example problems are also used as a verification of the software that has been developed to support the analysis. In all of these examples, “experimental” data is created from the integrator in the loop simulations. In the first example problem, ideal experimental data is collected, and compared to an ideal plant and ideal control. This problem was designed for verification purposes. The second problem, diode loss experiment with ideal plant and ideal control, was designed to test the ability to understand the prediction of the uncertainty with large uncertainty. The third example problem will test the ability to use the backstepping augmented control. The fourth problem will examine the parameter optimization routines. The fifth example problem will examine using the full model equations as described in Eqs. (5.1-5.5). The last problem will demonstrate the hardware filtering is likely to affect the results.

6.1 Ideal Experiment, Ideal Plant, Ideal Control (Example Problem 1)

The first problem is a comparison between an ideal model for the experiment with ideal control and a model with an ideal plant and ideal control. The goal of this example is to demonstrate that with no uncertainty in the model or control that the calculated model and control uncertainty is zero.
The differential equations implemented can be found in Eqs. (6.1 - 6.2).

\[
\frac{dV_o}{dt} = -\frac{I_o}{C} + \frac{V_i}{CV_o}, \\
\frac{dI_l}{dt} = \frac{V_i - V_o}{L} + \frac{V_o}{L}d_T.
\] (6.1) (6.2)

One issue found during this step involved producing results from the integrator in the loop model. In that model, it was necessary to add a zero order hold block (zoh block) to the model with an update rate of 14 kHz, because the inputs to the integrator out of the loop model were recorded at 14 kHz (see Fig. 6.1). The addition of the zero order hold block has destabilizing effects on the model, thus when the model is used for prediction care must be taken. The result of this change was that the model and control uncertainty of both differential equations were zero (see Tables 6.1 - 6.4).

As these tables will be used throughout the remainder of this dissertation, a brief discussion of the table information will be given. The table is composed of two columns, the first column

![Diagram](image)
corresponds to the measured uncertainty or tolerance interval for the first differential equation associated with Eq.(6.1) and the second column corresponds to the second differential Eq.(6.2). The three rows correspond to the three methods for tolerance interval calculation (see Appendix B for a complete description). The first row marked “Normal” is a tolerance interval calculation based on the normality assumption. The second row “Ordered Stat” is based on the downsampling method described in Appendix B. The last row is based on the maximum method as presented in Appendix B.

Table 6.1: Model Uncertainty Tolerance Intervals of Data Set $D_t$ for Example Problem 1.

<table>
<thead>
<tr>
<th></th>
<th>$\Delta m_1$ (V/s)</th>
<th>$\Delta m_2$ (A/s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normal</td>
<td>0.00</td>
<td>Normal 0.00</td>
</tr>
<tr>
<td>Ordered Stat</td>
<td>0.00</td>
<td>Ordered Stat 0.00</td>
</tr>
<tr>
<td>Max</td>
<td>0.00</td>
<td>Max 0.00</td>
</tr>
</tbody>
</table>

Table 6.2: Model Uncertainty Tolerance Intervals of Data Set $D_v$ for Example Problem 1.

<table>
<thead>
<tr>
<th></th>
<th>$\Delta m_1$ (V/s)</th>
<th>$\Delta m_2$ (A/s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normal</td>
<td>0.00</td>
<td>Normal 0.00</td>
</tr>
<tr>
<td>Ordered Stat</td>
<td>0.00</td>
<td>Ordered Stat 0.00</td>
</tr>
<tr>
<td>Max</td>
<td>0.00</td>
<td>Max 0.00</td>
</tr>
</tbody>
</table>

Table 6.3: Control Uncertainty Tolerance Intervals of Data Set $D_t$ for Example Problem 1.

<table>
<thead>
<tr>
<th></th>
<th>$\Delta c_1$ (V/s)</th>
<th>$\Delta c_2$ (A/s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normal</td>
<td>0.00</td>
<td>Normal 0.00</td>
</tr>
<tr>
<td>Ordered Stat</td>
<td>0.00</td>
<td>Ordered Stat 0.00</td>
</tr>
<tr>
<td>Max</td>
<td>0.00</td>
<td>Max 0.00</td>
</tr>
</tbody>
</table>
Table 6.4: Control Uncertainty Tolerance Intervals of Data Set $D_v$ for Example Problem 1.

<table>
<thead>
<tr>
<th></th>
<th>$\Delta c_1$ (V/s)</th>
<th>$\Delta c_2$ (A/s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normal</td>
<td>0.00</td>
<td>Normal 0.00</td>
</tr>
<tr>
<td>Ordered Stat</td>
<td>0.00</td>
<td>Ordered Stat 0.00</td>
</tr>
<tr>
<td>Max</td>
<td>0.00</td>
<td>Max 0.00</td>
</tr>
</tbody>
</table>

6.2 Diode Loss Experiment, Ideal Plant, Ideal Control (Example Problem 2)

In this example problem, the goals are to demonstrate the uncertainty measurements are similar to the analytical results, to demonstrate the uncertainty prediction capability developed under this work, and to demonstrate the potential of over or under estimation of the model uncertainty of an integrator in the loop model. It uses equations based on inclusion of a diode drop formulation in the experimental data with ideal control and an assumption of an ideal model and control for the simulation.

This experimental data generated was generated from the differential equations

\[
\frac{dV_o}{dt} = -\frac{I_o}{C} + \frac{V_i}{C(V_o + V_F)},
\]
\[
\frac{dI_l}{dt} = \frac{V_i - V_o - V_F}{L} + \frac{V_o + V_F}{L} d_T,
\]

while the simulation and control believe the system looks like

\[
\frac{dV_o}{dt} = -\frac{I_o}{C} + \frac{V_i}{C(V_o)},
\]
\[
\frac{dI_l}{dt} = \frac{V_i - V_o}{L} + \frac{V_o}{L} d_T,
\]

which results in an analytical uncertainty of

\[
\delta_{m1} = \frac{V_i}{C(V_o + V_F)} I_l - \frac{V_i}{C(V_o)} I_l,
\]
\[
\delta_{m2} = -\frac{V_F}{L} (d_T - 1).
\]
The first results of the analysis demonstrate the measured model uncertainty limit, $\Delta m_2$, is larger than the analytical results as seen in Table 6.5. In Table 6.5, the “Estimate” is based on a first order polynomial model that is described in Tables 6.6-6.9, the “Measured” column is based on the maximum estimation of the tolerance intervals found in Table 6.10, and the “Analytical Value” is calculated from Eqs. (6.3-6.4). The reason the measured uncertainty limit, $\Delta m_2$, is different than the estimate or the analytical value is because the tolerance interval used is the maximum tolerance interval which in this case (as seen in Table 6.10) is the tolerance interval associated with the normal assumption. Thus this discrepancy is not a problem with the uncertainty calculations from the data, but more an issue with the low number of points used in the tolerance interval calculations. The measured model uncertainty limit, $\Delta m_1$, matches well with the estimate and analytical values.

Table 6.5: Model Uncertainty Functional Estimate, Measured, and Analytical Uncertainty Comparisons for Data Set $D_t$ for Example Problem 2.

<table>
<thead>
<tr>
<th></th>
<th>Estimate</th>
<th>Measured</th>
<th>Analytical Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\Delta m_1$ (V/s)</td>
<td>35.5450</td>
<td>35.1613</td>
<td>35.1322</td>
</tr>
<tr>
<td>$\Delta m_2$ (A/s)</td>
<td>846.4344</td>
<td>918.3590</td>
<td>846.4264</td>
</tr>
</tbody>
</table>

Tables 6.6 and 6.8 display the coefficients of the polynomial that was constructed under the heading “Estimate”. Also calculated within those tables are the standard error for the parameter, the t-statistic indicating the likelihood that the coefficient is zero, and the p-value corresponding to the analysis of the t-value [8]. In Tables 6.7 and 6.9 are the regression diagnostics of the polynomial including the sample standard deviation, the mean of the output, the prediction error sum of squares (PRESS), coefficient of determination ($R^2$), adjusted coefficient of determination ($R^2_{adj}$), and coefficient of determination prediction ($R^2_{pred}$).
Table 6.6: δ₁, Model Uncertainty, Tuning Results for Data Set Dᵣ for Example Problem 2.

\begin{center}
\begin{tabular}{lcccr}

\hline
& Estimate & SE & tStat & pValue \\
\hline
(Intercept) & -26.4 & 0.0531 & -497 & 0 \\
Iₒ & -1.07e+03 & 0.898 & -1.19e+03 & 0 \\
Vₒ & 8.1 & 0.0176 & 459 & 0 \\
Iₒ : Vₒ & 75.3 & 0.109 & 693 & 0 \\
Vₒ² & -0.571 & 0.00143 & -400 & 0 \\
\hline
\end{tabular}
\end{center}

Table 6.7: Regression Diagnostics for δ₁ for Data Set Dᵣ for Example Problem 2.

\begin{center}
\begin{tabular}{lcc}

\hline
Std. Dev. & 0.5660 & R-Squared & 0.9956 \\
Mean & -26.5071 & R-Squared Adj & 0.9951 \\
PRESS & 8620.0619 & R-Squared Pred & 0.9956 \\
\hline
\end{tabular}
\end{center}

Table 6.8: δ₂, Model Uncertainty, Results for Data Set Dᵣ for Example Problem 2.

\begin{center}
\begin{tabular}{lcccr}

\hline
& Estimate & SE & tStat & pValue \\
\hline
(Intercept) & -1e+03 & 0.00188 & -5.32e+05 & 0 \\
dₜ & 1e+03 & 0.00373 & 2.68e+05 & 0 \\
\hline
\end{tabular}
\end{center}

Table 6.9: Regression Diagnostics for δ₂ for Data Set Dᵣ for Example Problem 2.

\begin{center}
\begin{tabular}{lcc}

\hline
Std. Dev. & 0.1069 & R-Squared & 1.0000 \\
Mean & -526.2590 & R-Squared Adj & 1.0000 \\
PRESS & 307.2587 & R-Squared Pred & 1.0000 \\
\hline
\end{tabular}
\end{center}

Table 6.10: Model Uncertainty Tolerance Intervals for Data Set Dᵣ for Example Problem 2.

\begin{center}
\begin{tabular}{lccc}

\hline
\multicolumn{2}{c}{Δ₁ (V/s)} & \multicolumn{2}{c}{Δ₂ (A/s)} \\
\hline
Normal & 35.09 & Normal & 918.36 \\
Ordered Stat & 35.16 & Ordered Stat & 846.23 \\
Max & 35.16 & Max & 918.36 \\
\hline
\end{tabular}
\end{center}

82
All the analysis so far has been with the data set $D_t$ (i.e., the tuning data). To predict data in a validation data set, $D_v$, an integrator in the loop model is used to generate state data for data set $D_{sv}$. Then, that data is used with model prediction Eq. (5.19) to predict the uncertainty limits of the validations data. Such a task was performed, along with evaluation of the model validation data ($D_v$) and the results are summarized in Table 6.11. Included in this table are the measured uncertainty from data set $D_v$, the estimate from the Lipschitz method based on Eq. (5.19), the uncertainty function estimate of the limit (labeled “Uncertainty Estimate”), along with a check on whether the Lipschitz estimate or the uncertainty function estimate of the model uncertainty limit met the risk criteria as specified in the requirements (see Section 5.5.2). The Lipschitz estimate has met the risk criteria (i.e., the number of misses was less than or equal to the required 7). The uncertainty limit based on the functional estimate did not meet the risk criteria.

Table 6.11: Model Hypothesis Testing for Example Problem 2.

<table>
<thead>
<tr>
<th></th>
<th>Lipschitz Estimate</th>
<th>Uncertainty Estimate</th>
<th>Measured Uncertainty</th>
<th># Misses Lipschitz Est. (Goal &lt;= 7)</th>
<th># Misses Uncertainty Est. (Goal &lt;= 7)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\Delta m_1$</td>
<td>73.81</td>
<td>35.49</td>
<td>56.46</td>
<td>0</td>
<td>21</td>
</tr>
<tr>
<td>$\Delta m_2$</td>
<td>968.33</td>
<td>876.01</td>
<td>829.84</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

A comparison with the analytical values is seen in Table 6.12. One takeaway from the results in this table is that the uncertainty limit estimate, $\Delta m_1$, does poorly (i.e., estimated value of 35.48 and an analytical value of 56.23) with a data set that was not used for tuning which can be typical of polynomial estimation and other black box modeling approaches. The theory behind this work is that the Lipschitz constant of the polynomial is sufficient information for the prediction.

The other item of interest is whether the data selected for validation could be considered strong validation as a function of the uncertainty models. Table 6.13 displays the distance calculation based...
Table 6.12: Uncertainty Function Estimate, Lipschitz Estimate, Measured Value, and Analytical Uncertainty Comparisons for Data Set $D_v$ for Example Problem 2.

<table>
<thead>
<tr>
<th></th>
<th>Uncertainty Estimate</th>
<th>Lipschitz Estimate</th>
<th>Measured Uncertainty</th>
<th>Analytical Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\Delta m_1$ (V/s)</td>
<td>35.49</td>
<td>73.81</td>
<td>56.46</td>
<td>56.23</td>
</tr>
<tr>
<td>$\Delta m_2$ (A/s)</td>
<td>876.01</td>
<td>968.33</td>
<td>829.84</td>
<td>829.97</td>
</tr>
</tbody>
</table>

on the tuning uncertainty models. This data indicates that the validation data is strong as seen in the more than 10x difference between the simulated to validation data sets ($D_{sv} \rightarrow D_v$) and the distance between the validated data set to the tuning data set ($D_v \rightarrow D_t$) (i.e., $||\Phi v_i - \Phi t_i|| > > ||\Phi v_i - \Phi sv_i||$).

Table 6.13: Euclidean Distance Between Data Sets Using $|| \ast ||^mt$ for Example Problem 2.

<table>
<thead>
<tr>
<th></th>
<th>Simulated to Tune</th>
<th>Validated to Tune</th>
<th>Simulated to Validated</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\Delta m_1$</td>
<td>19.3231</td>
<td>19.3679</td>
<td>1.0880</td>
</tr>
<tr>
<td>$\Delta m_2$</td>
<td>49.9750</td>
<td>49.4804</td>
<td>4.5765</td>
</tr>
</tbody>
</table>

Finally, the last point that will be made for this example problem, is the need for performing the uncertainty analysis with an integrator out of the loop model. This point is made by examining the measured uncertainty from the data generated from an integrator in the loop model (see Table 6.14) and comparing to the data generated from an integrator out of the loop model (see Table 6.10). The measured uncertainty is considerably better with the integrator in the loop model. It could have also likely had been greater or similar. The cause of the difference is the different state vectors used with the two models (see Figs. 6.2 and 6.3). Because the differential equations are evaluated with different state information the results of the comparison can be unpredictable.
Table 6.14: Model Uncertainty Tolerance Intervals for Integrator in the Loop Model with Data Set $D_t$ for Example Problem 2.

<table>
<thead>
<tr>
<th>$\Delta m_1$ (V/s)</th>
<th>$\Delta m_2$ (A/s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normal</td>
<td>4.96</td>
</tr>
<tr>
<td>Ordered Stat</td>
<td>4.64</td>
</tr>
<tr>
<td>Max</td>
<td>4.96</td>
</tr>
</tbody>
</table>

| Normal              | 4.20                |
| Ordered Stat        | 3.71                |
| Max                 | 4.20                |

Figure 6.2: Comparison of integrator in the loop vs. out of the loop for calculating the state inputs (output voltage state).
Figure 6.3: Comparison of integrator in the loop vs. out of the loop for calculating the state inputs (inductor current state).
6.3 Diode Loss Experiment, Diode Loss Plant, Ideal Control (Example Problem 3)

In this section, the experiment is the same as the previous experiment, it is assumed the plant is modeled correctly, and the control is the ideal control. In reality this is likely never to happen as the engineer would have updated the controller to match the plant, but this assumption is made to demonstrate and test the logic. The steps of the validation process to be examined here include the control tuning, control simulation testing, and control hypothesis testing.

As expected, there is a large uncertainty between the control model and the plant model as seen in Table 6.15. In fact the results demonstrate an exact match to the previous example problem as seen in Table 6.15.

Table 6.15: Control Uncertainty Tolerance Intervals for Data Set \( D_1 \) for Example Problem 3.

<table>
<thead>
<tr>
<th>( \Delta_{c_1} ) (V/s)</th>
<th>( \Delta_{c_2} ) (A/s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normal 35.09 Normal 918.36</td>
<td></td>
</tr>
<tr>
<td>Ordered Stat 35.16 Ordered Stat 846.23</td>
<td></td>
</tr>
<tr>
<td>Max 35.16 Max 918.36</td>
<td></td>
</tr>
</tbody>
</table>

However, there are intentionally created differences in the estimated uncertainty polynomials. There are limitations in the model form that can be used to generate a reasonable controller. For instance, in this problem having a term \( d_T \) in the first differential equation changes the form of the differential equation such that it is no longer in strict feedback form, and thus backstepping would not be possible. Although for the second differential equation, \( d_T \) would be allowed, it was decided to leave it out of the allowed signals for model identification to examine how this effects the results and demonstrate the potential for overestimation of the uncertainty. Also, \( V_i \) was intentionally left out of \( \delta_{c_1} \) to better understand what happens if \( \delta_{c_1} \) is poorly represented.
With the tuning data set, $D_t$, the following polynomial approximations of the uncertainties were developed (see Tables 6.16 - 6.19). With high values of the coefficients of determinations (e.g., $R^2$, $R^2_{adj}$, and $R^2_{pred}$), these are likely good models of the uncertainty, although not perfect of $\delta_{c2}$ as seen in the previous section.

Table 6.16: $\delta_{c1}$ Tuning Results for Data Set $D_t$ for Example Problem 3.

<table>
<thead>
<tr>
<th></th>
<th>Estimate</th>
<th>SE</th>
<th>tStat</th>
<th>pValue</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Intercept)</td>
<td>0.207</td>
<td>0.161</td>
<td>1.29</td>
<td>0.197</td>
</tr>
<tr>
<td>$I_t$</td>
<td>-13</td>
<td>1.08</td>
<td>-12</td>
<td>3.9e-33</td>
</tr>
<tr>
<td>$V_i$</td>
<td>-0.35</td>
<td>0.0448</td>
<td>-7.81</td>
<td>5.7e-15</td>
</tr>
<tr>
<td>$I_o$</td>
<td>-1.06e+03</td>
<td>1.58</td>
<td>-670</td>
<td>0</td>
</tr>
<tr>
<td>$I_t : V_i$</td>
<td>76.7</td>
<td>0.346</td>
<td>222</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 6.17: Regression Diagnostic Table for $\delta_{c1}$ for Data Set $D_t$ for Example Problem 3.

<table>
<thead>
<tr>
<th></th>
<th>Std. Dev.</th>
<th>R-Squared</th>
<th>R-Squared Adj</th>
<th>R-Squared Pred</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean</td>
<td>-26.5071</td>
<td>0.9568</td>
<td>0.9522</td>
<td>0.9568</td>
</tr>
<tr>
<td>PRESS</td>
<td>84963.5487</td>
<td>0.9568</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Testing of the augmented controller can be found in Table 6.20. The control uncertainty limits for $\Delta_{c1}$ were about the same between the two data sets. It would seem the augmented controller had little or no impact on the performance. However, without the augmented controller, the control uncertainty limits would have been higher.

The next step is creation of the control uncertainties, $\delta_{ci}$. The estimate of $\delta_{c1}$ has very low coefficient of determinations ($\sim$0.13) indicating that the augmented controller may have reduced much of the uncertainty not due to noise. The coefficient of determinations for $\delta_{c2}$ was high ($\sim$0.8).
Table 6.18: $\delta_{c2}$ Tuning Results for Data Set $D_t$ for Example Problem 3.

<table>
<thead>
<tr>
<th></th>
<th>Estimate</th>
<th>SE</th>
<th>tStat</th>
<th>pValue</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Intercept)</td>
<td>-133</td>
<td>4.31</td>
<td>-30.9</td>
<td>3.3e-206</td>
</tr>
<tr>
<td>$V_o$</td>
<td>18.8</td>
<td>0.603</td>
<td>31.2</td>
<td>2.11e-210</td>
</tr>
<tr>
<td>$I_o$</td>
<td>3.53e+03</td>
<td>33.7</td>
<td>105</td>
<td>0</td>
</tr>
<tr>
<td>$V_i$</td>
<td>-304</td>
<td>1.15</td>
<td>-264</td>
<td>0</td>
</tr>
<tr>
<td>$V_o : I_o$</td>
<td>-441</td>
<td>4.02</td>
<td>-110</td>
<td>0</td>
</tr>
<tr>
<td>$V_o : V_i$</td>
<td>21.8</td>
<td>0.161</td>
<td>136</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 6.19: Regression Diagnostic Table for $\delta_{c2}$ for Data Set $D_t$ for Example Problem 3.

<table>
<thead>
<tr>
<th></th>
<th>Std. Dev.</th>
<th>R-Squared</th>
<th>R-Squared Adj</th>
</tr>
</thead>
<tbody>
<tr>
<td>Std. Dev.</td>
<td>25.2635</td>
<td>0.9791</td>
<td></td>
</tr>
<tr>
<td>Mean</td>
<td>-526.2590</td>
<td>0.9760</td>
<td></td>
</tr>
<tr>
<td>PRESS</td>
<td>17171230.7683</td>
<td>0.9791</td>
<td></td>
</tr>
</tbody>
</table>

Table 6.20: Control Uncertainty Tolerance Intervals for Data Set $D_v$ for Example Problem 3.

<table>
<thead>
<tr>
<th></th>
<th>$\Delta_{c1}$ (V/s)</th>
<th>$\Delta_{c2}$ (A/s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normal</td>
<td>29.55</td>
<td>Normal</td>
</tr>
<tr>
<td>Ordered Stat</td>
<td>35.78</td>
<td>Ordered Stat</td>
</tr>
<tr>
<td>Max</td>
<td>35.78</td>
<td>Max</td>
</tr>
</tbody>
</table>

89
See Tables 6.21 and 6.23 for the polynomials created, and Tables 6.22 and 6.24 for the diagnostic information.

Table 6.21: $\delta_{c_1}$ Polynomial for Data Set $D_v$ for Example Problem 3.

<table>
<thead>
<tr>
<th></th>
<th>Estimate</th>
<th>SE</th>
<th>tStat</th>
<th>pValue</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Intercept)</td>
<td>-1.27</td>
<td>0.0381</td>
<td>-33.3</td>
<td>9.83e-242</td>
</tr>
<tr>
<td>$I_i$</td>
<td>32.2</td>
<td>0.33</td>
<td>97.5</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 6.22: Regression Diagnostic Table for $\delta_{c_1}$ from Data Set $D_v$ for Example Problem 3.

<table>
<thead>
<tr>
<th></th>
<th>Std. Dev.</th>
<th>R-Squared</th>
<th>R-Squared Adj</th>
<th>R-Squared Pred</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean</td>
<td>2.2348</td>
<td>0.1296</td>
<td>0.1296</td>
<td>0.1296</td>
</tr>
<tr>
<td>PRESS</td>
<td>657727.2403</td>
<td>0.1296</td>
<td>0.1296</td>
<td>0.1296</td>
</tr>
</tbody>
</table>

Table 6.23: $\delta_{c_2}$ Polynomial for Data Set $D_v$ for Example Problem 3.

<table>
<thead>
<tr>
<th></th>
<th>Estimate</th>
<th>SE</th>
<th>tStat</th>
<th>pValue</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Intercept)</td>
<td>109</td>
<td>0.787</td>
<td>139</td>
<td>0</td>
</tr>
<tr>
<td>$V_o$</td>
<td>-15.6</td>
<td>0.107</td>
<td>-146</td>
<td>0</td>
</tr>
<tr>
<td>$I_o$</td>
<td>-2.01e+03</td>
<td>13.7</td>
<td>-146</td>
<td>0</td>
</tr>
<tr>
<td>$\dot{I}_o$</td>
<td>-55.3</td>
<td>0.321</td>
<td>-172</td>
<td>0</td>
</tr>
<tr>
<td>$V_i$</td>
<td>14.5</td>
<td>0.0683</td>
<td>213</td>
<td>0</td>
</tr>
<tr>
<td>$V_o : I_o$</td>
<td>227</td>
<td>1.96</td>
<td>116</td>
<td>0</td>
</tr>
<tr>
<td>$V_o : \dot{I}_o$</td>
<td>7.98</td>
<td>0.0463</td>
<td>172</td>
<td>0</td>
</tr>
</tbody>
</table>

In the next step, control algorithm prediction, estimates of the control uncertainty limits are made. The results and comparison of those prediction is found in Table 6.25. The column “Lipschitz Estimate” is the estimate for the simulated future data as seen from Eq. (5.20). The column
Table 6.24: Regression Diagnostic Table for $\delta_{c2}$ from Data Set $D_v$ for Example Problem 3.

| Std. Dev.  | 4.9758 | R-Squared   | 0.7991 |
| Mean       | 29.1726 | R-Squared Adj | 0.7847 |
| PRESS      | 1582899.1251 | R-Squared Pred | 0.7990 |

“Uncertainty Estimate” is based on the uncertainty functions $\delta_{c1}$ and $\delta_{c2}$. The measured column is based on the maximum tolerance interval calculation. As can be seen the estimates based on the uncertainty functions are above and below the measured and the Lipschitz estimates are both above the measured. Unfortunately, the Lipschitz estimate for $\Delta_{c2}$ is very high, and that is a result of the large coefficients in the estimate to $\delta_{c2}$ as seen in Table 6.23. The problem is exacerbated due to the estimate used for the distance function as seen in Table 6.28. In that table it is shown that the distance between the validation data set and the tuning data set is much larger than the distance from the simulated data set to the validation data set (i.e., strong validation data). This last point is one of the reasons for introducing the estimates as opposed to the proofs in the Lipschitz calculations of the future data set uncertainty limits. Had the actual distance from the simulated to the validation data, the Lipschitz estimate would have been approximately 140 A/s. Obviously the actual distance can not be used in an estimation situation, and thus the over-estimate may happen here. Table 6.26 does show the hypothesis that the Lipschitz estimate is a good representation has not been falsified.

Table 6.25: Control Uncertainty Function Estimate, Lipschitz Estimate, and Measurement Uncertainty Comparison for Data Set $D_s$ for Example Problem 3.

<table>
<thead>
<tr>
<th></th>
<th>Uncertainty Estimate</th>
<th>Lipschitz Estimate</th>
<th>Measured Uncertainty</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\Delta_{c1}$ (V/s)</td>
<td>7.5973</td>
<td>35.8067</td>
<td>21.2599</td>
</tr>
<tr>
<td>$\Delta_{c2}$ (A/s)</td>
<td>88.9484</td>
<td>406.3952</td>
<td>55.0266</td>
</tr>
</tbody>
</table>
Table 6.26: Hypothesis Testing Results of the Control Uncertainty Limits for Data Set \( D_s \) for Example Problem 3.

<table>
<thead>
<tr>
<th>( \Delta_c1 )</th>
<th>( \Delta_c2 )</th>
<th>( \Delta_c2 )</th>
<th>( \Delta_c2 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lipschitz Estimate</td>
<td>Uncertainty Estimate</td>
<td>Measured Uncertainty</td>
<td># Misses Lipschitz Est. (Goal ( \leq 7 ))</td>
</tr>
<tr>
<td>35.81</td>
<td>7.60</td>
<td>21.26</td>
<td>0</td>
</tr>
<tr>
<td>406.40</td>
<td>88.95</td>
<td>55.03</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 6.27: Control Uncertainty Tolerance Intervals for Data Set \( D_s \) for Example Problem 3.

<table>
<thead>
<tr>
<th>( \Delta_c1 ) (V/s)</th>
<th>( \Delta_c2 ) (A/s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normal 20.34</td>
<td>Normal 52.44</td>
</tr>
<tr>
<td>Ordered Stat 21.26</td>
<td>Ordered Stat 55.03</td>
</tr>
<tr>
<td>Max 21.26</td>
<td>Max 55.03</td>
</tr>
</tbody>
</table>

One item that stood out with this example problem is the validation data and simulated validation data were not similar for purposes of evaluating the uncertainty \( \delta_{c1} \) (see Table 6.28). The distance from the simulated data set to the validated data set was much larger than the distance from the validated data set to the tuning data set. However, one also should note the model was not a good model as indicated in the coefficients of determination.

Table 6.28: Euclidean distance between data sets (Control Distances) \( || * ||^{cu} \) for Example Problem 3.

<table>
<thead>
<tr>
<th>( \Delta_c1 )</th>
<th>( \Delta_c2 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Validated to Tune 0.0264</td>
<td>Simulated to Tuned 0.0265</td>
</tr>
<tr>
<td>142.7294</td>
<td>140.6424</td>
</tr>
</tbody>
</table>
6.4 Diode Loss Experiment (with parameter changes), Diode Loss Plant, Diode Loss Control (Example Problem 4)

In this section, data is taken that includes the diode loss as mentioned in the previous section but includes parameter changes to the capacitance, inductance, the duty cycle gain and offset, and the diode voltage drop parameter. The first test of this configuration will be to test the ability of the optimization routine to estimate the new parameters and to examine how the Lipschitz analysis works with what should be relatively small uncertainty.

Values used in the experimental setup along with values initially used, and the optimized parameter set are displayed in Table 6.29. Most of the parameter values were found without a problem, the lone exception was the inductance value. Smaller optimization sets (for example, three variables) did not exhibit the problem. Likely the issue is that a local minimum was found. The optimization routine used was fmincon with the $J_x$ cost function, and is one reason that a genetic optimization algorithm will be looked at for the experimental results.

Table 6.29: Parameter Optimization for Fourth Example Problem.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Experimental</th>
<th>Initial Values</th>
<th>Optimized Simulation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inductor (uH)</td>
<td>333.0</td>
<td>327.0</td>
<td>330.0218</td>
</tr>
<tr>
<td>Capacitor (uF)</td>
<td>80.00</td>
<td>94.20</td>
<td>79.9970</td>
</tr>
<tr>
<td>Diode Voltage Drop (V)</td>
<td>0.310</td>
<td>0.3270</td>
<td>0.3099998</td>
</tr>
<tr>
<td>Duty Cycle Gain</td>
<td>0.97</td>
<td>1.000</td>
<td>0.9700003</td>
</tr>
<tr>
<td>Duty Cycle Offset</td>
<td>0.03</td>
<td>0</td>
<td>29.9999e-003</td>
</tr>
</tbody>
</table>

With this set of parameters, the measured uncertainty was very small as seen in Table 6.30. This low level of measured uncertainty (i.e., only noise is present) that led to model uncertainties that are not accurate. In fact $\delta_{m1}$ was only a constant and had coefficient of determinations near 0, and $\delta_{m2}$ has coefficient of determinations of 0.84 (see Tables 6.31 and 6.32).
Table 6.30: Model Uncertainty Tolerance Intervals of Data Set $D_t$ for Example Problem 4.

<table>
<thead>
<tr>
<th></th>
<th>$\Delta m_1$ (V/s)</th>
<th>$\Delta m_2$ (A/s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normal</td>
<td>0.00</td>
<td>1.76</td>
</tr>
<tr>
<td>Ordered Stat</td>
<td>0.00</td>
<td>1.80</td>
</tr>
<tr>
<td>Max</td>
<td>0.00</td>
<td>1.80</td>
</tr>
</tbody>
</table>

Table 6.31: Regression Diagnostic Table for $\delta m_1$ from Data Set $D_t$ for Example Problem 4

<table>
<thead>
<tr>
<th></th>
<th>Std. Dev.</th>
<th>R-Squared</th>
<th>R-Squared Adj</th>
<th>R-Squared Pred</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean</td>
<td>-0.0000</td>
<td>-0.0000</td>
<td>0.0312</td>
<td>-0.0000</td>
</tr>
<tr>
<td>PRESS</td>
<td>0.0000</td>
<td>0.8354</td>
<td>0.8241</td>
<td>0.8350</td>
</tr>
</tbody>
</table>

The Lipschitz estimate and direct estimation were rather poor as seen in Table 6.33 and 6.34. Both estimation techniques underestimated the measured values and both hypothesis that the model was good was demonstrated as false (i.e., the number of misses was higher than the critical number) except for the Lipschitz mode uncertainty limit $\Delta m_2$. Obviously these are relatively small values so this would not be an issue, in fact, what it says is that the plant model used is quite good.

Because the model was in a form that was usable by the control, and because the control was updated based on the parameter optimization, the control uncertainty is zero through tuning and validation so it will not be presented here.

Table 6.32: Regression Diagnostic Table for $\delta m_2$ from Data Set $D_t$ for Example Problem 4.

<table>
<thead>
<tr>
<th></th>
<th>Std. Dev.</th>
<th>R-Squared</th>
<th>R-Squared Adj</th>
<th>R-Squared Pred</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean</td>
<td>0.0310</td>
<td>0.8354</td>
<td>0.8241</td>
<td>0.8350</td>
</tr>
<tr>
<td>PRESS</td>
<td>25.8722</td>
<td>0.8354</td>
<td>0.8350</td>
<td></td>
</tr>
</tbody>
</table>
Table 6.33: Uncertainty Function Estimate, Lipschitz Estimate, and Measured Comparison using Data Set $D_v$ for Example Problem 4.

<table>
<thead>
<tr>
<th></th>
<th>Uncertainty Estimate</th>
<th>Lipschitz Estimate</th>
<th>Measured Uncertainty</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\Delta m_1$ (V/s)</td>
<td>0.0000</td>
<td>0.0009</td>
<td>0.0142</td>
</tr>
<tr>
<td>$\Delta m_2$ (A/s)</td>
<td>1.3877</td>
<td>2.8665</td>
<td>3.2410</td>
</tr>
</tbody>
</table>

Table 6.34: Hypothesis Test Results of Model Uncertainty for Data Set $D_v$ for Example Problem 4.

<table>
<thead>
<tr>
<th></th>
<th>Lipschitz Estimate</th>
<th>Uncertainty Estimate</th>
<th>Measured Uncertainty</th>
<th># Misses Lipschitz Est. (Goal $&lt;=$ 7)</th>
<th># Misses Uncertainty Est. (Goal $&lt;=$ 7)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\Delta m_1$</td>
<td>0.00</td>
<td>0.00</td>
<td>0.01</td>
<td>68</td>
<td>76</td>
</tr>
<tr>
<td>$\Delta m_2$</td>
<td>2.87</td>
<td>1.39</td>
<td>3.24</td>
<td>6</td>
<td>40</td>
</tr>
</tbody>
</table>

6.5 Full Plant Experiment (No Time Delay), Full Plant, Full Control (Example Problem 5)

For the next set of experiments, the plant is in such a form that the control can not be updated to match (i.e., the plant model is not in strict feedback form). This experiment includes the full plant model as found in Eqs. (5.1 - 5.4) but does not include the controller delay as will be discussed in Section 7.2.

One point made is the ability of the parameter optimization to find only a local minimum similar to issues found in the previous section. It will also be demonstrated that the control uncertainty for the inductor current equation (see Eq. (5.2)) is zero if the delay between the duty cycle output and the plant is removed.

The optimization here was performed with fmincon with $J_x$ as the cost function. The results of the optimization in Table 6.35 shows that most parameters match within 1% except for the capacitor.
ESR which varied by about 3%. This indicates two things, one that there are likely many local min-
imums and the optimization metric has low sensitivity to the ESR of the capacitor. An optimization
routine like a genetic algorithm may offer better performance as will be demonstrated in Chapter

VIII

Table 6.35: Parameter Optimization for Example Problem 5, Difference between the Experimental
Parameter Values and the Optimized Parameter Values.

<table>
<thead>
<tr>
<th></th>
<th>Experimental Value</th>
<th>Initial Condition</th>
<th>Optimized Value</th>
<th>% Diff</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inductor (uH)</td>
<td>327.0000</td>
<td>334.0000</td>
<td>330.7350</td>
<td>1.14%</td>
</tr>
<tr>
<td>Inductor ESR (Ohm)</td>
<td>0.6460</td>
<td>0.6200</td>
<td>0.6459</td>
<td>-0.01%</td>
</tr>
<tr>
<td>Capacitor (uF)</td>
<td>94.2000</td>
<td>95.0000</td>
<td>93.7801</td>
<td>-0.45%</td>
</tr>
<tr>
<td>Capacitor ESR (Ohm)</td>
<td>0.3230</td>
<td>0.2763</td>
<td>0.3120</td>
<td>-3.41%</td>
</tr>
<tr>
<td>Diode Voltage Drop (V)</td>
<td>0.3270</td>
<td>0.3000</td>
<td>0.3270</td>
<td>0.00%</td>
</tr>
<tr>
<td>Diode Resistance (Ohm)</td>
<td>1.0000</td>
<td>1.0949</td>
<td>1.0001</td>
<td>0.01%</td>
</tr>
<tr>
<td>Diode 2nd Order Term</td>
<td>-4.1000</td>
<td>-4.2440</td>
<td>-4.0995</td>
<td>-0.01%</td>
</tr>
<tr>
<td>Duty Cycle Gain</td>
<td>1.0000</td>
<td>0.9128</td>
<td>1.0000</td>
<td>0.00%</td>
</tr>
<tr>
<td>Duty Cycle Offset</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.00%</td>
</tr>
<tr>
<td>Jx</td>
<td>2.13E-12</td>
<td>8.53E+02</td>
<td>6.59E-03</td>
<td></td>
</tr>
</tbody>
</table>

Tables 6.36 and 6.37 show that the model uncertainty limits are relatively small with the model
uncertainty limits calculated from the validation data set as slightly larger. The same issue occurs
as the previous example problem, the predicted limits for the model validation data set are proven
to be false via the hypothesis testing in Table 6.38 except for the Lipschitz estimate of \( \Delta m_2 \). In this
situation and the last, the model for \( \delta m_1 \) is a constant and thus the Lipschitz estimate will have little
or no impact.

Without a delay the control uncertainty, \( \delta c_2 \), is zero as seen in Table 6.39. In the control algo-
rithm validation chapter, Chapter IX, the control uncertainty limit, \( \Delta c_2 \) will be shown to be about
300 A/s. The uncertainty \( \delta c_1 \) is impacted by the delay as will also be seen in Chapter IX. Also
Table 6.36: Model Uncertainty Tolerance Intervals for Data Set $D_t$ for Example Problem 5.

<table>
<thead>
<tr>
<th>$\Delta m_1$ (V/s)</th>
<th>$\Delta m_2$ (A/s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normal 1.16</td>
<td>Normal 2.46</td>
</tr>
<tr>
<td>Ordered Stat 1.12</td>
<td>Ordered Stat 2.54</td>
</tr>
<tr>
<td>Max 1.16</td>
<td>Max 2.54</td>
</tr>
</tbody>
</table>

Table 6.37: Model Uncertainty Tolerance Intervals for Data Set $D_v$ for Example Problem 5.

<table>
<thead>
<tr>
<th>$\Delta m_1$ V/s</th>
<th>$\Delta m_2$ A/s</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normal 5.24</td>
<td>Normal 3.96</td>
</tr>
<tr>
<td>Ordered Stat 5.26</td>
<td>Ordered Stat 4.58</td>
</tr>
<tr>
<td>Max 5.26</td>
<td>Max 4.58</td>
</tr>
</tbody>
</table>

Table 6.38: Hypothesis Testing Results of Model Uncertainty for Data Set $D_v$ for Example Problem 5.

<table>
<thead>
<tr>
<th>Lipschitz Estimate</th>
<th>Uncertainty Estimate</th>
<th>Measured Uncertainty</th>
<th># Misses Lipschitz Est. (Goal $&lt;=$ 7)</th>
<th># Misses Uncertainty Est. (Goal $&lt;=$ 7)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\Delta m_1$ 1.16</td>
<td>0.00</td>
<td>5.26</td>
<td>56</td>
<td>76</td>
</tr>
<tr>
<td>$\Delta m_2$ 4.38</td>
<td>2.10</td>
<td>4.58</td>
<td>1</td>
<td>28</td>
</tr>
</tbody>
</table>
note in previous simulation work, a controller having a loop rate of 20 kHz had lower values of the control uncertainty limit $\Delta_{c_1}$ [77].

Table 6.39: Control Uncertainty Tolerance Intervals for Data Set $D_s$ for Example Problem 5.

<table>
<thead>
<tr>
<th>$\Delta_{c_1}$ (V/s)</th>
<th>$\Delta_{c_2}$ (A/s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normal</td>
<td>24.35</td>
</tr>
<tr>
<td>Ordered Stat</td>
<td>24.58</td>
</tr>
<tr>
<td>Max</td>
<td>24.58</td>
</tr>
</tbody>
</table>

Finally, there is larger uncertainty in the control, $\delta_{c_1}$, for four test points in particular (see Fig. 6.4). Those points are points 4, 14, 15, and 30. Typically points 4, 14, and 30 are the worst and point 15 is nearly as uncertain. The points 4, 14, and 30 are replicated tests that require the highest average duty cycle as the commanded voltage is at 9.5 V and the input voltage is low at 3 V for points 4, 14, and 30 and 3.5 V for point 15. With the switch in resistance occurring from two resistors to one resistor. Further investigation is necessary to understand the root cause, but it is likely due to either the $d_T$ simplification for the controller implementation or possibly $\dot{d}_T$ in the calculation of $\dot{V}_o$.

6.6 Simple Experiment (with filtered data), Simple Plant, Simple Control (Example Problem 6)

Up until now, all the data used as inputs to the states of the open loop simulation has been the unfiltered outputs of the model. The experimental setup will have filtered outputs that make use of a second order Butterworth filter with a bandwidth of 10 kHz. This experiment, plant, and control is based on the ideal model similar to example one except for the outputs of the simple experiment are filtered in the same manner as the experiment (i.e., a 10 kHz second order Butterworth filter). Because the plant and control relation are the same, the control uncertainty is zero. The only analysis presented is for the model uncertainty $\delta_{m_1}$ and $\delta_{m_2}$.
The measured model uncertainties for the tuning data set are displayed in Table 6.40 and the measured model uncertainties for the validation data set are displayed in Table 6.41. The bottom line is that the best the uncertainty measurement is likely to be for the validation data set is $\Delta_{m1} = 29.94 \text{ V/s}$ and $\Delta_{m2} = 11.47 \text{ A/s}$. Future work should consider means of reverse filtering the experimental data to recover the dynamics of the unfiltered data. The intent of the experimental filtering is to avoid anti-aliasing of the switching frequency of 100 kHz, so simply removing the filter is not an option.
Table 6.40: Model Uncertainty Tolerance Intervals for Data Set $D_t$ for Example Problem 6.

<table>
<thead>
<tr>
<th></th>
<th>$\Delta m_1$ (V/s)</th>
<th>$\Delta m_2$ (A/s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normal</td>
<td>9.03</td>
<td>Normal</td>
</tr>
<tr>
<td>Normal</td>
<td></td>
<td>0.57</td>
</tr>
<tr>
<td>Ordered Stat</td>
<td>8.68</td>
<td>Ordered Stat</td>
</tr>
<tr>
<td>Ordered Stat</td>
<td></td>
<td>0.59</td>
</tr>
<tr>
<td>Max</td>
<td>9.03</td>
<td>Max</td>
</tr>
<tr>
<td>Max</td>
<td></td>
<td>0.59</td>
</tr>
</tbody>
</table>

Table 6.41: Model Uncertainty Tolerance Intervals for Data Set $D_v$ for Example Problem 6.

<table>
<thead>
<tr>
<th></th>
<th>$\Delta m_1$ (V/s)</th>
<th>$\Delta m_2$ (A/s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normal</td>
<td>24.76</td>
<td>Normal</td>
</tr>
<tr>
<td>Normal</td>
<td></td>
<td>7.44</td>
</tr>
<tr>
<td>Ordered Stat</td>
<td>29.94</td>
<td>Ordered Stat</td>
</tr>
<tr>
<td>Ordered Stat</td>
<td></td>
<td>11.47</td>
</tr>
<tr>
<td>Max</td>
<td>29.94</td>
<td>Max</td>
</tr>
<tr>
<td>Max</td>
<td></td>
<td>11.47</td>
</tr>
</tbody>
</table>
CHAPTER VII

HARDWARE DESIGN AND UNCERTAINTY ANALYSIS

7.1 DC-DC Boost Converter Circuit Design

The original goals for the boost circuit was that it have a proven output voltage between 8.5 and 9.5 V (with an average value of 9 V) with an input voltage between 6 and 8 V and loads of 100 Ω or 200 Ω. However, due to the measurement uncertainty, this level of voltage regulation (proven) was not possible. So instead, it was decided to track to 7 V (Lyapunov stability analysis proof would show the output voltage would be bounded to stay between 5 and 9.5 V) with an input voltage between 3 and 4 V. Tracking to 7 V was chosen because the data acquisition system analog to digital card range was ±10 V.

To meet the original goals, a design process developed from [64] was used. It was determined that the boost circuit would operate in continuous conduction mode and have a switching frequency of 100 kHz. As a result of the design process, the following components were chosen. To meet the output voltage requirements with a voltage input varying between 6 and 8 volts, the minimum inductance was found to be 296 µH and thus an inductor of 330 µH was chosen with 10% tolerance. The capacitance was chosen primarily based on the required voltage ripple. With a voltage ripple of 0.1 V, the minimum capacitance was 2.3 µF with a series resistance maximum of 0.3 Ω. Originally a capacitor value of 3.3 µF with a 10% tolerance was chosen, but it was found to have a series
resistance in the 2 Ω range. So a larger capacitance was chosen (≈100 µF) with a lower series resistance of 0.5 Ω. With the selection of 100 µF and series resistance of 0.5 Ω the ripple requirement was met. With these selections the natural frequency response of the boost circuit was about 660 Hz. The requirements for the diode were that it be able to withstand a reverse voltage of up to 10 V, and a current rating of 0.09 A. A MBR2535CTL diode was chosen because it has a 12.5 A current limit with a 35 V reverse voltage rating. The n-MOSFET selection had similar requirements and an IRF540 n-MOSFET was chosen. It has a rating of 100 V drain to source and 28 A maximum drain current. Choice of these particular components were driven more by availability in the lab and ensuring they met the requirements. Sense resistors were also obtained to measure the input and output current of the boost circuit. The sense resistors chosen were 0.47 Ω with a tolerance of 5 %. Finally for loading of the circuit, 200 Ω resistors with tolerances of 10 % were chosen. To switch between one or two resistors, another IRF540 n-MOSFET was used. Finally, TC4420 gate drive chips were selected to drive the IRF540 n-MOSFETs. Figure 7.1 depicts the placement of the parts.

The subsequent goals were also met with these part choices if one increases the ripple voltage requirement to 0.15 V as opposed to 0.1 V. So it was decided to move forward. Note, the frequency of the ripple voltage is the same as the switching frequency and is beyond the frequency range of interest. Also note, the low pass filter on the data acquisition boards is set to a second order Butterworth filter with a cutoff frequency of 10 kHz and will remove this ripple from the data.

### 7.2 Experimental Setup

An experimental setup is required to validate the accuracy of the simulation. As such, a dc-dc boost converter was built based on the components selected during design, a data acquisition system was obtained, and a control system was obtained to allow for digital control of the circuit. A pictorial layout of the system can be found in Fig. 7.2.
Figure 7.1: Pictorial layout of the experimental setup.

Figure 7.2: Pictorial layout of the experimental setup.
For control and data acquisition, a real-time PCI based system from SpeedGoat was used. This system has an optimized BIOS and hardware set for real-time data control and acquisition. The system has an Intel Core i7 processor running at 3.5 GHz with a National Instruments (NI) PCIe-6259 (16 bit A/D with 1.25 MS/s) card for analog output. The operating system used is National Instruments RT PharLap version 13.1. The model and data acquisition was setup through Veristand version 2013. The NI PCIe-6259 has a channel that can be used for PWM output so it was chosen as the card for control. The control rate for the system was 14 kHz to meet real-time execution. For data acquisition and control input filtering, a NI PXIe-4300 board inside a NI PXIe-1062Q rack was used. The NI PXIe-1062Q rack communicates to the real-time SpeedGoat controller via fiber. The NI PXIe-4300 board was used because it has selectable anti-aliasing filter capability with a two-pole Butterworth filter that has a cut-off frequency of 10 kHz.

Changes to the board were made to slow down the transition from one resistor to two resistors, because the transition was occurring faster than the data acquisition system could collect more than one point, thus missing likely dynamics. To facilitate the slowing of the transition, a filter was added between the gate drive and the IRF540 gate input, the effective filter was a first order with a cutoff frequency of 80 Hz and is depicted in Fig. 7.1. The filter was an R-C filter with a resistance of $2 \, \text{k} \Omega$ and a capacitance of 1 $\mu F$.

Originally the controller was designed with a 20 kHz loop rate, but that proved impossible to meet real-time and thus reduced to 14 kHz. Also, of note through testing it was found that there was a time delay due to NI software on the output of the controller of approximately one loop (i.e., $1/(14 \, \text{kHz})$). This proved to be one of the limiting factors in providing good control of the system. Future work will include acquiring a controller / and software capable of faster loop-times and without the delay.
7.3 Uncertainty Analysis

7.3.1 Component Uncertainty Analysis

The goal of the component uncertainty analysis is to bound the parameter identification problem. The component uncertainty analysis will identify likely ranges of the parameters of interest so that during identification, the search area can be limited. During this analysis, only uninstalled values will be measured. For example, in [17] it was demonstrated that the resistance in the RLC circuit under test was not only determined by the resistor but the resistance due to the inductor, the wiring, and breadboard. To limit these additional installed effects, measurement of the inductor’s resistance is included and a printed circuit board was created to reduce the dependence on wires and the breadboard.

The types of component uncertainties that should be measured/obtained for such a system could include manufacturing uncertainty, measurement uncertainty, measurement system uncertainty, and for some components frequency uncertainty (or ideal model uncertainty (e.g., the model used here for the capacitor does not include stray capacitance, thus there is variation in capacitance and resistance as a function of frequency)). For this work, the manufacturing variation will not be accounted for and it will be assumed that the control is being developed for one system. If the goal was to develop controls across different parts, then uncertainty propagation of the components would be used in addition to several circuits would need to be tested. For the resistors, capacitors, and inductors an LCR meter (HP 4284A) was used to measure each of the available components. While measuring the capacitance and inductance, series resistance were also measured. Further, the inductance, capacitance, and series resistance measurements were taken at various frequencies to understand the variation due to frequency. The variation due to frequency was a dominant uncertainty for the capacitor and inductor and was most likely due to stray inductances in the capacitors and stray capacitance in the inductors.
In Table 7.1 the capacitance and series resistance is detailed as a function of frequency. As can be seen up to the 100 kHz frequency there is some change in the capacitance and some change in the series resistance, these changes are primarily due to the stray inductance. If one wanted to include these effects in the model then additional physics (i.e., the stray inductance) would need to be modeled. It was felt at this time, the additional physics were not necessary. It was also thought it would be beneficial to allow for the system identification to better pick the dominant capacitance. Similar effects were seen with the inductor (see Table 7.2), though in this case, only the series resistance changed. These are the capacitor and inductor used in the circuit that was tested.

Table 7.1: Capacitor B24 capacitance and series resistance as a function of frequency.

<table>
<thead>
<tr>
<th>Frequency (Hz)</th>
<th>Cs (µF)</th>
<th>Rs (Ω)</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>94.6</td>
<td>0.497995</td>
</tr>
<tr>
<td>1000</td>
<td>92.0</td>
<td>0.357636</td>
</tr>
<tr>
<td>5000</td>
<td>87.2</td>
<td>0.317365</td>
</tr>
<tr>
<td>20000</td>
<td>76.6</td>
<td>0.298715</td>
</tr>
<tr>
<td>100000</td>
<td>64.6</td>
<td>0.275241</td>
</tr>
</tbody>
</table>

Table 7.2: Inductor B15 inductance and series resistance as a function of frequency.

<table>
<thead>
<tr>
<th>Frequency (Hz)</th>
<th>Ls (µH)</th>
<th>Rs (Ω)</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>333</td>
<td>0.63236</td>
</tr>
<tr>
<td>1000</td>
<td>333</td>
<td>0.634919</td>
</tr>
<tr>
<td>50000</td>
<td>332</td>
<td>1.3658</td>
</tr>
<tr>
<td>100000</td>
<td>331</td>
<td>2.8685</td>
</tr>
</tbody>
</table>

For the diodes, a separate circuit was setup to measure the forward voltage drop, on resistance, and higher order effects. Initially only the forward voltage drop and on resistance were calculated,
later it was found more resolution was needed for the diode model (i.e., higher order terms). A comparison of second order (forward voltage drop and on resistance) was performed with a second order model (see Fig. 7.3). For this measurement a voltage supply (HP 6032A) applied positive voltage across the anode end and a resistance in series with the diode (see Fig. 7.4). The voltage drop across the diode and the resistor were measured with the DAQ system and plotted. Based on the data, the forward voltage and on resistance were estimated for each diode (see Fig. 7.3). Note, Fig. 7.3 is of diode D1 and diode D6 was used for all experimentation. Also note, each package of the MBR2535CTL is composed of two diodes, and all work uses both diodes from the package.

Note this analysis is based on constant current through the diode. Further investigation is needed to understand if the second order model is sufficient for a switched diode voltage drop.

Figure 7.3: Plot of I-V characteristic of diodes and model of diode (Diode - D1).
Characterization of the n-MOSFET was done in a similar manner as the diode, though an additional voltage is applied to the gate pin (see Fig. 7.5). Results of the n-MOSFET testing are summarized in Table 7.3.

For all of the components (resistors, inductors, diodes, and n-MOSFETS), one component was chosen and 50 repeated measurements were made to understand the measurement uncertainty. The results of the measurement variation can be found in Table 7.3 under the columns “Lower Limit std” and “Upper Limit std”, this is the standard deviation of the measurement variation at either the upper limit or lower limit of the parameter value (e.g., since the capacitance of the capacitor varied by nearly 50% due to changes in test frequencies, a measurement variation study was done for the low frequency data and for the high frequency data). The parameter upper limit was set to the mean (if it exists otherwise the mean of the highest values) plus three standard deviations of the measurement variation plus the accuracy variation. The accuracy variation is the reported accuracy variation of the measurement device (i.e., its system uncertainty). It is assumed that the since the
systemic measurement accuracy of the data acquisition boards used for the n-MOSFET and diodes is considerably smaller than the random uncertainty, it was not included in the calculation. Three standard deviations was chosen to ensure that the parts true value was captured and to allow for some change due to possible installed effects. The component used in the variation study was not the component that was used in the experiment but it is expected that the measurement uncertainty was similar between the components. These components had been used in a generation one of the circuit board. The values used for the optimization studies can be found in Table 7.4 where the upper limit values were rounded up and the lower limit values were rounded down to account for some more of the potential installed effects.

### 7.3.2 Measurement Uncertainty Analysis

To make use of an experimental setup it is important to understand the uncertainty in measurements obtained from the system. The NI PXIe-4300 card was evaluated through a process that was
### Table 7.3: Uncertainty Analysis of Components for DC-DC Boost Circuit.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Upper Limit</th>
<th>Lower Limit</th>
<th>Mean</th>
<th>Lower Limit std</th>
<th>Upper Limit std</th>
<th>Low Freq. Accuracy</th>
<th>High Freq. Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inductor (uH)</td>
<td>333.8</td>
<td>329.7</td>
<td>N/A</td>
<td>7.276E-02</td>
<td>3.697E-02</td>
<td>0.32%</td>
<td>0.11%</td>
</tr>
<tr>
<td>Inductor ESR (Ohm)</td>
<td>2.8933</td>
<td>0.6267</td>
<td>N/A</td>
<td>1.197E-03</td>
<td>7.210E-03</td>
<td>0.32%</td>
<td>0.11%</td>
</tr>
<tr>
<td>Capacitor (uF)</td>
<td>94.7867</td>
<td>53.6813</td>
<td>N/A</td>
<td>1.966E+00</td>
<td>4.914E-02</td>
<td>0.09%</td>
<td>7.73%</td>
</tr>
<tr>
<td>Capacitor ESR (Ohm)</td>
<td>0.5232</td>
<td>0.2349</td>
<td>N/A</td>
<td>6.368E-03</td>
<td>8.267E-03</td>
<td>0.09%</td>
<td>7.73%</td>
</tr>
<tr>
<td>Sense Resistor 1 (Ohm)</td>
<td>0.4771</td>
<td>0.4630</td>
<td>0.4701</td>
<td>8.124E-04</td>
<td>8.124E-04</td>
<td>0.98%</td>
<td>0.98%</td>
</tr>
<tr>
<td>Sense Resistor 2 (Ohm)</td>
<td>0.4760</td>
<td>0.4619</td>
<td>0.4690</td>
<td>8.124E-04</td>
<td>8.124E-04</td>
<td>0.98%</td>
<td>0.98%</td>
</tr>
<tr>
<td>MOSFET Resistance (Ohm)</td>
<td>0.0417</td>
<td>0.0374</td>
<td>0.0396</td>
<td>7.261E-04</td>
<td>7.261E-04</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Diode Voltage Drop (V)</td>
<td>0.3203</td>
<td>0.3060</td>
<td>0.3131</td>
<td>2.387E-03</td>
<td>2.387E-03</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Diode Resistance (Ohm)</td>
<td>1.0713</td>
<td>0.9775</td>
<td>1.0244</td>
<td>1.564E-02</td>
<td>1.564E-02</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Diode 2nd Order Term</td>
<td>-4.0559</td>
<td>-4.3699</td>
<td>-4.2129</td>
<td>5.233E-02</td>
<td>5.233E-02</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Load Resistor 1 (Ohm)</td>
<td>200.2</td>
<td>199.0</td>
<td>199.6</td>
<td>8.066E-04</td>
<td>8.066E-04</td>
<td>0.29%</td>
<td>0.14%</td>
</tr>
<tr>
<td>Load Resistor 2 (Ohm)</td>
<td>200.4</td>
<td>199.2</td>
<td>199.8</td>
<td>8.0661E-04</td>
<td>8.0661E-04</td>
<td>0.29%</td>
<td>0.14%</td>
</tr>
</tbody>
</table>

### Table 7.4: Range of Values used for Optimization Studies of Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>ub</th>
<th>lb</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inductor (uH)</td>
<td>334.00</td>
<td>329.00</td>
</tr>
<tr>
<td>Inductor ESR (Ohm)</td>
<td>2.90</td>
<td>0.62</td>
</tr>
<tr>
<td>Capacitor (uF)</td>
<td>95.00</td>
<td>53.00</td>
</tr>
<tr>
<td>Capacitor ESR (Ohm)</td>
<td>0.53</td>
<td>0.23</td>
</tr>
<tr>
<td>Diode Voltage Drop (V)</td>
<td>0.33</td>
<td>0.30</td>
</tr>
<tr>
<td>Diode Resistance (Ohm)</td>
<td>1.10</td>
<td>0.97</td>
</tr>
<tr>
<td>Diode 2nd Order Term (V/A²)</td>
<td>-4.00</td>
<td>-4.50</td>
</tr>
</tbody>
</table>
recently developed to obtain the total dc uncertainty of the card (NI PXIe-4300) and the breakout board (NI TB-4300). This process is discussed in [62],[63]. From this process the total dc uncertainty of the card was found to be ± 360 µV (2 σ), so any measurements taken with this system represent a value ± 360 µV from the true value with 95% confidence. Note, the systematic uncertainty was ± 57 µV and the random uncertainty was about ± 356 µV. Also note this value is for the voltage ranges from ± 10 V. When measuring the inductor current or output current through the sense resistor, the voltage range used was ± 1 V. This range had total dc uncertainty of ± 50 µV, with a systematic uncertainty of ± 12 µV. All uncertainties here are with 2 σ confidence.

The inductor and output current are measured using a sense resistor. The uncertainty for the inductor current is ± 2.2 mA and the uncertainty on the output current is ± 0.94 mA. This uncertainty is in part due to the uncertainty of the voltage measurement and in part due to the uncertainty in the resistance of the resistor. The following develops the equations used to calculate that uncertainty

\[
\begin{align*}
s_F &= s_r \sqrt{1 + \frac{1}{N_r}} \\
s_b &= s_v \sqrt{1 + \frac{1}{N_v}} \\
b_r &= \frac{0.98\% \bar{r}_x}{2} \\
b_v &= \frac{4.7e - 6}{2} V \\
\frac{\partial I}{\partial V} &= \frac{1}{\bar{r}_x} \\
\frac{\partial I}{\partial R} &= \frac{\bar{v}_{rx}}{\bar{r}_x^2} \\
\frac{\partial I}{\partial R} &= \sqrt{\left(s_r \frac{\partial I}{\partial R}\right)^2 + \left(s_v \frac{\partial I}{\partial V}\right)^2} \\
t &= t_{1-\alpha/2,\nu} \sqrt{s_t^2 + b_t^2} \\
u_q &= \frac{t_{1-\alpha/2,\nu} \sqrt{s_t^2 + b_t^2}}{ \sqrt{\frac{s_r^2}{N_r} + \frac{s_v^2}{N_v}}} \\
\end{align*}
\]

where \(x\) is either 1 or 2 depending on which sense resistor, \(s_r\) is the sample standard deviation from the resistor measurement random uncertainty, \(s_v\) is the random uncertainty associated with the DAQ
card, \( b_r \) is the systematic uncertainty from the LCR meter, \( b_v \) is the systematic uncertainty from the DAQ card, \( N_r \) is the number of resistance measurements of the resistors, \( \bar{r}_x \) is the sample average resistance, \( N_v \) is the number of measurements taken to determine the uncertainty of the DAQ card, and \( u_q \) is the uncertainty. The values for this calculation can be found in Table 7.5. Implementation code for these equations can be found in Appendix H.1.

### Table 7.5: Values used in the current uncertainty calculation.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \bar{r}_1 )</td>
<td>0.4701 Ω</td>
</tr>
<tr>
<td>( \bar{r}_2 )</td>
<td>0.4690 Ω</td>
</tr>
<tr>
<td>( s_r )</td>
<td>812.4 µΩ</td>
</tr>
<tr>
<td>( s_v )</td>
<td>24.45 µV</td>
</tr>
<tr>
<td>( N_r )</td>
<td>50</td>
</tr>
<tr>
<td>( N_v )</td>
<td>35</td>
</tr>
<tr>
<td>( b_r )</td>
<td>0.49 % ( \bar{r}_x ) Ω</td>
</tr>
<tr>
<td>( b_v )</td>
<td>5.883 µV</td>
</tr>
</tbody>
</table>

Typically in uncertainty quantification, the random uncertainty is associated with a confidence interval of the measurement. Here, a prediction interval is used because only single data points will be used in the future [63].

#### 7.3.3 Derivative Calculation Uncertainty Analysis

Due to the random uncertainty in the voltage and current measurements, the derivative calculation also has some level of random uncertainty. This section will describe the Monte-Carlo analysis that was performed to estimate the derivative calculation uncertainty. Because the derivative calculation also contained a low pass filter, direct methods similar to what was used for the current calculation could not be used. Note, the derivative calculation uncertainty is not affected by systematic uncertainty because it is a comparison of signals.
The calculation used for the derivative estimate is based on the equation

\[ \dot{x} = \frac{\lambda p}{p + \lambda} x(t), \]  
\[ (7.1) \]

then using the bilinear transformation, the discrete implementation is the following

\[ \dot{x}(1) = \frac{x(2) - x(1)}{t_s} \]

\[ \dot{x}(k + 1) = \frac{2x(k + 1)}{2\alpha ft_s} - \frac{2x(k)}{2\alpha ft_s} - \dot{x}(k) \frac{t_s}{2\alpha ft_s} \]

where \( t_s \) is the sample time and \( \alpha_f \) is the parameter indicating cutoff frequency where \( \alpha_f = \frac{1}{\lambda} \).

For the work presented here, the cutoff frequency was 5 kHz with \( \alpha_f = 3.1831 \times 10^{-5} \). The above code is implemented in Appendix \[E.4\].

For the estimation, an ideal sine wave was used for comparison before and after addition of random uncertainty. So first a signal was constructed as a sine wave with a frequency of 5 kHz (a similar test was done with a frequency of 1 kHz and similar results were obtained), then the derivative calculation was done on the signal. Then 5000 test signals were created as sine waves with a frequency of 5 kHz with normally distributed noise with a variance of 31.7 nV\(^2\) (i.e., \((356 \mu V / 2)^2 = 31.7 \text{nV}\)) for the voltage estimation and 1.21 \(\mu A^2\) (i.e., \((2.2 \text{mA}/2)^2 = 1.21 \mu A^2\)) for the current estimation. The variance used, was based on the random uncertainty found in Section \[7.3.2\].

For each test, a mean error, the sample standard deviation, and a 95% random uncertainty level was calculated. The 95% random uncertainty level was estimated by adding the mean error to two times the sample standard deviation. The results showed the output voltage derivative uncertainty was 7.7 V/s and the inductor current was 47.5 A/s. See Appendix \[H.2\] for the implementation.
CHAPTER VIII

MODEL VALIDATION

The next two chapters will present the results of the dc-dc boost circuit model analysis based on the methods described in Chapter V. Three different approaches at generating parameter sets will be compared against the baseline set that was created based on the analysis from Chapter VII. Throughout the analysis it will be seen that the three methods of parameter optimization perform relatively the same. The process of estimating the model uncertainty limits via the two Lipschitz methods will be validated for all three new parameter sets and baseline data set. At the end, it will be demonstrated that the three new parameter sets will predict the model uncertainty limits of the final data set within the risk requirements.

8.1 System and/or Parameter Identification

The changes as a result of the system identification discussed in Section 5.5.3 and use of $V_o$ as the state variable are presented here in differential equation form. The following is the updated
plant differential equations

\[ d_T(t) = d_T g u_c(t - T_D) + d_T o, \]  
\( r = r_l + r_{s n s 1} + d_T r_{d s} - R_F (d_T - 1), \)

\[ \dot{V}_c = -I_o + I_l \frac{(d_T - 1)}{C}, \]

\[ \dot{i}_i = \frac{V_i - I_l r - I_o r_{s n s 2} + V_F (d_T - 1) + V_o (d_T - 1) + D_F I_l^2 (d_T - 1)}{L}, \]

\[ \dot{V}_o = \dot{V}_c - I_o r_{s n s 2} - r_c \left( \dot{I}_o + \dot{I}_l (d_T - 1) + I_l d_T \right), \]  
(8.2)

where \( T_D \) is the delay in the control, \( d_T g \) and \( d_T o \) are representative of the additional dynamics of the n-MOSFET.

Three different approaches of finding parameter values will be compared in this section. The different optimization approaches used were fmincon, as previously discussed in Section 5.5.3 that uses both cost functions \( J_x \) defined in Eq.(5.17) and \( \dot{J}_x \) defined in Eq.(5.18) and a genetic algorithm (GA) that used the cost function \( J_x \). The parameters found as a result of the optimization can be found in Table 8.1. The cost function evaluations for each of the resultant parameter sets can be found in Table 8.2. The first two rows are the cost functions found in Eqs.(5.17 and 5.18) respectively, and the last two rows are maximum values of the model uncertainty for the tuning data set. Note, the last two columns are not necessarily the tolerance intervals, but the actual maximum value.

Summarizing these results, fmincon showed use of \( J_x \) as the cost function performed better than the cost function based on \( \dot{J}_x \). One likely explanation is that the flatness of the optimization space along with the number of local minimums increased when using the derivative of the output error. Finally, the genetic algorithm optimization method did show improvements over the fmincon in some metrics; however, the computational time used was considerably more than fmincon. For
the results presented, the two fmincon optimizations took approximately 12 hours each compared
to 72 hours for the genetic algorithm.

Table 8.1: Parameter Sets used In the Model Validation Comparison.

<table>
<thead>
<tr>
<th></th>
<th>Baseline</th>
<th>Fmincon w/ $J_x$</th>
<th>GA</th>
<th>Fmincon w/ $J_{\dot{x}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inductor (uH)</td>
<td>331.000</td>
<td>333.997</td>
<td>332.929</td>
<td>334.000</td>
</tr>
<tr>
<td>Inductor ESR (Ohm)</td>
<td>0.646</td>
<td>1.151</td>
<td>1.000</td>
<td>1.226</td>
</tr>
<tr>
<td>Capacitor (uF)</td>
<td>94.160</td>
<td>94.999</td>
<td>94.921</td>
<td>95.000</td>
</tr>
<tr>
<td>Capacitor ESR (Ohm)</td>
<td>0.323</td>
<td>0.230</td>
<td>0.232</td>
<td>0.230</td>
</tr>
<tr>
<td>Diode Voltage Drop (V)</td>
<td>0.315</td>
<td>0.301</td>
<td>0.328</td>
<td>0.311</td>
</tr>
<tr>
<td>Diode Resistance (Ohm)</td>
<td>1.029</td>
<td>1.064</td>
<td>1.090</td>
<td>1.092</td>
</tr>
<tr>
<td>Diode 2nd Order Term (V/A^2)</td>
<td>-4.266</td>
<td>-4.248</td>
<td>-4.463</td>
<td>-4.139</td>
</tr>
<tr>
<td>Duty Cycle Gain</td>
<td>1.000</td>
<td>0.989</td>
<td>0.973</td>
<td>0.987</td>
</tr>
<tr>
<td>Duty Cycle Offset</td>
<td>0.000</td>
<td>0.027</td>
<td>0.034</td>
<td>0.031</td>
</tr>
<tr>
<td>Control Delay ($\mu$S)</td>
<td>71.4</td>
<td>71.4</td>
<td>71.4</td>
<td>71.4</td>
</tr>
</tbody>
</table>

Table 8.2: Comparison of Different Optimization Routines.

<table>
<thead>
<tr>
<th></th>
<th>Baseline</th>
<th>Fmincon w/ $J_x$</th>
<th>GA</th>
<th>Fmincon w/ $J_{\dot{x}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$J_x$</td>
<td>289.0</td>
<td>44.1</td>
<td>37.9</td>
<td>55.7</td>
</tr>
<tr>
<td>$J_{\dot{x}}$</td>
<td>10918.3</td>
<td>1375.7</td>
<td>1535.1</td>
<td>1679.2</td>
</tr>
<tr>
<td>$\Delta m_1$ (V/s)</td>
<td>123.9</td>
<td>68.3</td>
<td>50.0</td>
<td>79.5</td>
</tr>
<tr>
<td>$\Delta m_2$ (A/s)</td>
<td>483.8</td>
<td>121.7</td>
<td>106.7</td>
<td>114.1</td>
</tr>
</tbody>
</table>

Next, creation of uncertainty estimates for the model will be done to determine if the risk re-
quirements will likely be met. It will be shown, the model accuracy requirements that were set forth
will likely be met (i.e., $\Delta m_1 < 100$ V/s and $\Delta m_2 < 150$ A/s). Unfortunately, in the next chapter, the
control uncertainty limit, $\Delta c_2$, will be found to be greater than 300 A/s, thus the requirements will
not be met. The total uncertainty limit, $\Delta 2$, was only 300 A/s. Only the model uncertainty estimates
based on the parameter sets from fmincon with $J_x$ (referred to as just fmincon) and the genetic algorithm will be presented here. The model uncertainty estimates for the parameter set from fmincon with $J_x$ can be found in Appendix I. The techniques used in generating the uncertainty estimates can be found in Appendix E.

8.1.1 Model Uncertainty Analysis for the Fmincon Model

The uncertainty analysis for the fmincon parameter set found in Tables 8.3, 8.4, 8.5, and 8.6 indicate the uncertainty model, $\delta_m 1$, explains a majority of the signal, but the model $\delta_m 2$ does not. The coefficient of determinations for $\delta_m 2$ are less than 0.5, but this low coefficient of determination is primarily due to the low signal to noise ratio. In the Hardware and Software Setup, Chapter VII, the estimate two standard deviation level of noise for $I_l$ was 47.5 A/s, while the estimated two standard deviation of the model is slightly higher at 56 A/s. Thus most of the expected available signal is represented. While the coefficient of determination for $\delta_m 1$ is at 0.8 with a two standard deviation of noise estimate of 11 V/s which is higher than the expected 7.7 V/s indicating additional signal content is likely available. A comparison was done between the maximum value of $\delta_{mi}$ over data set $D_t$ and the calculated value of $\Delta_{mi}$ based on the tolerance intervals, and the maximum value of the $\delta_{mi}$ were less than one half of the calculated value of $\Delta_{mi}$ (see Table 8.7). The key, though is whether the derivatives were captured, and that evaluation will be done later with the validation data set.

Table 8.3: $\delta_m 1$ Polynomial for Data Set $D_t$ for Model Fmincon w/ $J_x$ Cost Function.

<table>
<thead>
<tr>
<th></th>
<th>Estimate</th>
<th>SE</th>
<th>tStat</th>
<th>pValue</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Intercept)</td>
<td>43.6</td>
<td>0.272</td>
<td>160</td>
<td>0</td>
</tr>
<tr>
<td>$I_o$</td>
<td>-1.13e+03</td>
<td>6.71</td>
<td>-169</td>
<td>0</td>
</tr>
<tr>
<td>$d_T$</td>
<td>-106</td>
<td>0.494</td>
<td>-215</td>
<td>0</td>
</tr>
<tr>
<td>$I_o : d_T$</td>
<td>2.44e+03</td>
<td>10.4</td>
<td>234</td>
<td>0</td>
</tr>
</tbody>
</table>
Table 8.4: Regression Diagnostic Table for $\delta_{m1}$ for Data Set $D_t$ for Model Fmincon w/ $J_x$ Cost Function.

<table>
<thead>
<tr>
<th></th>
<th>Std. Dev.</th>
<th>R-Squared</th>
<th>R-Squared Adj</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean</td>
<td>2.2816</td>
<td>0.8154</td>
<td>0.8026</td>
</tr>
<tr>
<td>PRESS</td>
<td>803971.6578</td>
<td>0.8153</td>
<td></td>
</tr>
</tbody>
</table>

Table 8.5: $\delta_{m2}$ Polynomial for Data Set $D_t$ for Model Fmincon w/ $J_x$ Cost Function.

<table>
<thead>
<tr>
<th></th>
<th>Estimate</th>
<th>SE</th>
<th>tStat</th>
<th>pValue</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Intercept)</td>
<td>298</td>
<td>2.2</td>
<td>135</td>
<td>0</td>
</tr>
<tr>
<td>$d_T$</td>
<td>-570</td>
<td>5.52</td>
<td>-103</td>
<td>0</td>
</tr>
<tr>
<td>$V_i$</td>
<td>-40.4</td>
<td>0.447</td>
<td>-90.3</td>
<td>0</td>
</tr>
<tr>
<td>$d_T^2$</td>
<td>502</td>
<td>5.82</td>
<td>86.2</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 8.6: Regression Diagnostic Table for $\delta_{m2}$ for Data Set $D_t$ for Model Fmincon w/ $J_x$ Cost Function.

<table>
<thead>
<tr>
<th></th>
<th>Std. Dev.</th>
<th>R-Squared</th>
<th>R-Squared Adj</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean</td>
<td>12.0790</td>
<td>0.4191</td>
<td>0.3790</td>
</tr>
<tr>
<td>PRESS</td>
<td>21512874.7706</td>
<td>0.4190</td>
<td></td>
</tr>
</tbody>
</table>

8.1.2 Model Uncertainty Analysis for the Genetic Algorithm Model

The analysis of the GA model showed the uncertainty models did not explain all the data (see Tables 8.8, 8.9, 8.10, and 8.11). In fact for $\delta_{m2}$ the model was only a constant and had coefficient of determinations near 0. This result indicates most of the “information” from the data has already been captured in the model. The uncertainty estimates as compared to the measured uncertainty again
Table 8.7: Model Uncertainty Estimate and Measured Comparisons for Data Set $D_t$ for Model Fmincon w/ $J_x$ Cost Function.

<table>
<thead>
<tr>
<th></th>
<th>Estimate</th>
<th>Measured</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\Delta m_1$ (V/s)</td>
<td>28.1515</td>
<td>68.2657</td>
</tr>
<tr>
<td>$\Delta m_2$ (A/s)</td>
<td>59.7440</td>
<td>127.8170</td>
</tr>
</tbody>
</table>

indicate that these uncertainty estimates do not perform a good job of estimation of the maximum uncertainty.

Table 8.8: $\delta m_1$ Polynomial for Data Set $D_t$ for Model GA.

<table>
<thead>
<tr>
<th></th>
<th>Estimate</th>
<th>SE</th>
<th>tStat</th>
<th>pValue</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Intercept)</td>
<td>32.9</td>
<td>0.273</td>
<td>121</td>
<td>0</td>
</tr>
<tr>
<td>$I_o$</td>
<td>-867</td>
<td>6.72</td>
<td>-129</td>
<td>0</td>
</tr>
<tr>
<td>$d_T$</td>
<td>-73.7</td>
<td>0.497</td>
<td>-148</td>
<td>0</td>
</tr>
<tr>
<td>$I_o : d_T$</td>
<td>1.67e+03</td>
<td>10.5</td>
<td>159</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 8.9: Regression Diagnostic Table for $\delta m_1$ for Data Set $D_t$ for Model GA.

<table>
<thead>
<tr>
<th></th>
<th>Std. Dev.</th>
<th>R-Squared</th>
<th>R-Squared Adj</th>
<th>R-Squared Pred</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean</td>
<td>-0.8504</td>
<td>0.5761</td>
<td>0.5468</td>
<td></td>
</tr>
<tr>
<td>PRESS</td>
<td>789230</td>
<td>0.5760</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

8.2 Model Prediction and Hypothesis Testing

The last two steps were combined for analysis purposes, so that the predictions could be evaluated. Before beginning to make predictions, the distance between the data sets needs to be calculated. For prediction purposes the distance from $D_{st} \rightarrow D_t$ is calculated for each of the
Table 8.10: $\delta_{m2}$ Polynomial for Data Set $D_t$ for Model GA.

<table>
<thead>
<tr>
<th></th>
<th>Estimate</th>
<th>SE</th>
<th>tStat</th>
<th>pValue</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Intercept)</td>
<td>12.6</td>
<td>0.253</td>
<td>49.9</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 8.11: Regression Diagnostic Table for $\delta_{m2}$ for Data Set $D_t$ for Model GA.

<table>
<thead>
<tr>
<th></th>
<th>Std. Dev.</th>
<th>R-Squared</th>
<th>R-Squared Adj</th>
<th>R-Squared Pred</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean</td>
<td>12.6324</td>
<td>0.0000</td>
<td>0.0313</td>
<td>-0.0000</td>
</tr>
<tr>
<td>PRESS</td>
<td>46311734</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

models and can be found in Table 8.13. As can be seen the strong validation assumption that $||\Phi_{vi} - \Phi_{ti}||_{mt} > > ||\Phi_{svi} - \Phi_{sti}||_{mt}$ and $||\Phi_{svi} - \Phi_{sti}||_{mt} \approx ||\Phi_{vi} - \Phi_{ti}||_{mt}$ was a good assumption. The lone exception was the calculation based on the baseline model of the second differential equation (i.e., $||\Phi_{svi} - \Phi_{sti}||_{2} > ||\Phi_{vi} - \Phi_{ti}||_{2}$). The leading indicators of the problem is that the coefficient of determination of $\delta_{m2}$ was 0.28 and the standard deviation was 97 indicating low explanation of the uncertainty of the model (see Table 8.14). Compare those results to those for the fmincon model in Table 8.6. Also, notice that the GA model also had issues as seen in Table 8.11.

Table 8.12: Model Uncertainty Function Estimate and Measured Comparisons for Data Set $D_t$ for Model GA.

| Δm1 (V/s) | 13.0473 | 49.9461 |
| Δm2 (A/s) | 12.6324 | 117.1932 |

120
In regards to the distance table, the distance for the GA model for the second differential equation is zero. This evaluation to zero is due to the fact the uncertainty model in this case is a constant, and thus the Lipschitz constant for a equation of just an offset is zero.

Using the estimation from Eq. (5.19) uncertainty estimate limits for the model validation data set were calculated and can be found in Table 8.15. They are labeled as $\hat{\Delta}_{mi}(D_v)_L$. The other estimate $\hat{\Delta}_{mi}(D_v)_u$ is calculated directly from the uncertainty equations $\delta_i$. This table data indicates the Lipschitz uncertainty estimate limits are valid (i.e., meet the risk requirements as specified of less than or equal to seven misses). Examining the uncertainty estimates $\hat{\Delta}_{mi}(D_v)_u$ one can see that all of those estimates other than the estimate $\hat{\Delta}_{m1}(D_v)_u$ from the baseline were inadequate based on the risk allowed.

Another item to note, is that the model uncertainty limit, $\Delta_{m2}(D_v)$, for the baseline model is quite large, nearly five times larger than the other models, indicating these models improvements did
help. Also note, that even though the GA model performed better during tuning than the fmincon model, it does not perform better during the validation as both model uncertainty limits $\Delta_m(D_v)$ for the fmincon model are better.

Table 8.15: Hypothesis Testing Results of Model Uncertainty for Data Set $D_v$.

<table>
<thead>
<tr>
<th></th>
<th>Baseline</th>
<th>Fmincon w/ $J_x$</th>
<th>GA</th>
<th>Fmincon w/ $J_x$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\Delta m_1(D_v)$</td>
<td>125.2</td>
<td>68.27</td>
<td>49.95</td>
<td>79.45</td>
</tr>
<tr>
<td>$\Delta m_2(D_v)$</td>
<td>123.9</td>
<td>127.82</td>
<td>117.19</td>
<td>131.22</td>
</tr>
<tr>
<td>$\hat{\Delta} m_1(D_v)_L$</td>
<td>415.2603</td>
<td>92.57</td>
<td>67.6206</td>
<td>108.04</td>
</tr>
<tr>
<td>$\hat{\Delta} m_2(D_v)_L$</td>
<td>543.3789</td>
<td>200.66</td>
<td>117.19</td>
<td>228.62</td>
</tr>
<tr>
<td>$\Delta m_1(D_v)_u$</td>
<td>110.4414</td>
<td>23.51</td>
<td>10.12</td>
<td>34.8</td>
</tr>
<tr>
<td>$\hat{\Delta} m_2(D_v)_u$</td>
<td>446.94</td>
<td>52.28</td>
<td>12.63</td>
<td>48.72</td>
</tr>
<tr>
<td>$\hat{\Delta} m_1(D_v)_L$ misses</td>
<td>0</td>
<td>0</td>
<td>7</td>
<td>0</td>
</tr>
<tr>
<td>$\hat{\Delta} m_2(D_v)_L$ misses</td>
<td>3</td>
<td>0</td>
<td>7</td>
<td>0</td>
</tr>
<tr>
<td>$\hat{\Delta} m_1(D_v)_u$ misses</td>
<td>1</td>
<td>74</td>
<td>76</td>
<td>50</td>
</tr>
<tr>
<td>$\hat{\Delta} m_2(D_v)_u$ misses</td>
<td>12</td>
<td>35</td>
<td>76</td>
<td>58</td>
</tr>
</tbody>
</table>

At the end of this stage, the uncertainty estimates functions are recalculated based on the new validation data along with the tuning data. The results for the fmincon and GA model are posted here in Table 8.16 - 8.23. The coefficients of determinations for all models except $\delta m_2$ for the GA model are reduced compared to those from the tuning data set. It is likely these uncertainty models are more appropriate estimation of the uncertainty throughout this range as opposed to the tuning models. Further, in Section 7.3.3, the random uncertainty associated with the derivative measurement of the output voltage was calculated as 7.7 V/s (two standard deviations) and the inductor current was calculated as 47.5 A/s (two standard deviations), so any model that predicts the noise to be less than this are over-predicting, and any models that predict the noise to this level are likely at the limits of the coefficient of determination. For example, in Table 8.19, the estimated
standard deviation the noise is 20.9 A/s, with a known noise of 23.25 A/s this is a fairly close representation, and it is unlikely a model with a coefficient of determination larger than 0.4 will be found.

Table 8.16: $\delta_{m1}$ Polynomial for Data Set $D_v$ for Model fmincon w/ $J_x$ cost function.

<table>
<thead>
<tr>
<th></th>
<th>Estimate</th>
<th>SE</th>
<th>tStat</th>
<th>pValue</th>
</tr>
</thead>
<tbody>
<tr>
<td>$(\text{Intercept})$</td>
<td>59</td>
<td>0.32</td>
<td>185</td>
<td>0</td>
</tr>
<tr>
<td>$V_i$</td>
<td>-7.2</td>
<td>0.0652</td>
<td>-110</td>
<td>0</td>
</tr>
<tr>
<td>$I_o$</td>
<td>-761</td>
<td>5.02</td>
<td>-152</td>
<td>0</td>
</tr>
<tr>
<td>$d_T$</td>
<td>-99.5</td>
<td>0.4</td>
<td>-249</td>
<td>0</td>
</tr>
<tr>
<td>$I_o : d_T$</td>
<td>1.99e+03</td>
<td>8.29</td>
<td>240</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 8.17: Regression Diagnostic Table for $\delta_{m1}$ for Data Set $D_v$ for Model fmincon w/ $J_x$ cost function.

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Std. Dev.</td>
<td>5.4243</td>
<td>0.7322</td>
<td></td>
</tr>
<tr>
<td>Mean</td>
<td>-1.2479</td>
<td>0.7245</td>
<td></td>
</tr>
<tr>
<td>PRESS</td>
<td>2672032</td>
<td>0.7322</td>
<td></td>
</tr>
</tbody>
</table>

Table 8.18: $\delta_{m2}$ Polynomial for Data Set $D_v$ for Model fmincon w/ $J_x$ cost function.

<table>
<thead>
<tr>
<th></th>
<th>Estimate</th>
<th>SE</th>
<th>tStat</th>
<th>pValue</th>
</tr>
</thead>
<tbody>
<tr>
<td>$(\text{Intercept})$</td>
<td>277</td>
<td>1.18</td>
<td>236</td>
<td>0</td>
</tr>
<tr>
<td>$d_T$</td>
<td>-550</td>
<td>3.15</td>
<td>-175</td>
<td>0</td>
</tr>
<tr>
<td>$V_i$</td>
<td>-37.1</td>
<td>0.237</td>
<td>-156</td>
<td>0</td>
</tr>
<tr>
<td>$d_T^2$</td>
<td>507</td>
<td>3.34</td>
<td>152</td>
<td>0</td>
</tr>
</tbody>
</table>
Table 8.19: Regression Diagnostic Table for $\delta_{m2}$ for Data Set $D_v$ for Model fmincon w/ $J_x$ cost function.

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Std. Dev.</td>
<td>20.8713</td>
<td>R-Squared</td>
<td>0.3899</td>
</tr>
<tr>
<td>Mean</td>
<td>4.0196</td>
<td>R-Squared Adj</td>
<td>0.3783</td>
</tr>
<tr>
<td>PRESS</td>
<td>39560852</td>
<td>R-Squared Pred</td>
<td>0.3898</td>
</tr>
</tbody>
</table>

Table 8.20: $\delta_{m1}$ Polynomial for Data Set $D_v$ for Model GA.

<table>
<thead>
<tr>
<th>Estimate</th>
<th>SE</th>
<th>tStat</th>
<th>pValue</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Intercept)</td>
<td>43.9</td>
<td>0.329</td>
<td>134</td>
</tr>
<tr>
<td>$V_i$</td>
<td>-6.31</td>
<td>0.0666</td>
<td>-94.7</td>
</tr>
<tr>
<td>$I_o$</td>
<td>-482</td>
<td>5.19</td>
<td>-92.8</td>
</tr>
<tr>
<td>$d_T$</td>
<td>-65.5</td>
<td>0.415</td>
<td>-158</td>
</tr>
<tr>
<td>$I_o : d_T$</td>
<td>1.22e+03</td>
<td>8.61</td>
<td>142</td>
</tr>
</tbody>
</table>

Table 8.21: Regression Diagnostic Table for $\delta_{m1}$ for Data Set $D_v$ for Model GA.

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Std. Dev.</td>
<td>5.5389</td>
<td>R-Squared</td>
<td>0.4743</td>
</tr>
<tr>
<td>Mean</td>
<td>-2.8687</td>
<td>R-Squared Adj</td>
<td>0.4591</td>
</tr>
<tr>
<td>PRESS</td>
<td>2786053</td>
<td>R-Squared Pred</td>
<td>0.4742</td>
</tr>
</tbody>
</table>

Table 8.22: $\delta_{m2}$ Polynomial for Data Set $D_v$ for Model GA.

<table>
<thead>
<tr>
<th>Estimate</th>
<th>SE</th>
<th>tStat</th>
<th>pValue</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Intercept)</td>
<td>178</td>
<td>1.03</td>
<td>173</td>
</tr>
<tr>
<td>$V_i$</td>
<td>-48.5</td>
<td>0.292</td>
<td>-166</td>
</tr>
</tbody>
</table>

Table 8.23: Regression Diagnostic Table for $\delta_{m2}$ for Data Set $D_v$ for Model GA.

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Std. Dev.</td>
<td>29.3446</td>
<td>R-Squared</td>
<td>0.2330</td>
</tr>
<tr>
<td>Mean</td>
<td>8.2074</td>
<td>R-Squared Adj</td>
<td>0.2330</td>
</tr>
<tr>
<td>PRESS</td>
<td>78198918</td>
<td>R-Squared Pred</td>
<td>0.2330</td>
</tr>
</tbody>
</table>

124
8.3 Future Data Prediction and Future Data Analysis

Once the model and the Lipschitz predictions have been validated, one can then apply it to other data sets with a prescribed level of confidence. For this problem the future data set $D_f$ is nearly identical to that taken for validation $D_v$ making this a slightly easier problem then the hypothesis testing where the tuning and validation data were more different. Future, work should explore the ability of this method to predict data more different from the validation data set to understand the limitations.

So before the final data was taken, predictions were made based on the three parameter sets. The baseline parameter set was not used due to the fact the model uncertainty limit, $\Delta m^2(D_v)$, was so large compared to the other models. Before the estimate is made, the distance calculation were determined and reported in Table 8.24. The distance from $D_v$ to $D_t$ was the largest for both differential equations for all three parameter sets. So the assumptions that $||\Phi_{vi} - \Phi_{ti}||_{1i}^{mv} > ||\Phi_{fi} - \Phi_{vi}||_{i}^{mv}$ was true for all three parameter sets. Further, $||\Phi_{si} - \Phi_{vi}||_{i}^{mv}$ is slightly lower than $||\Phi_{fi} - \Phi_{vi}||_{i}^{mv}$. It would be preferred that it is slightly higher. In fact the reason the distance $||\Phi_{si} - \Phi_{vi}||_{i}^{mv}$ is so small for the GA model is that the uncertainty function is only a function of $V_i$, which will be very similar for the two data sets (see Table 8.22).

Table 8.24: Distance Calculation used for Prediction of Data Set $D_f$.

<table>
<thead>
<tr>
<th></th>
<th>Fmincon w/ $J_x$</th>
<th>GA</th>
<th>Fmincon w/ $J_{\Phi}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$</td>
<td></td>
<td>\Phi_{vi} - \Phi_{ti}</td>
<td></td>
</tr>
<tr>
<td>$</td>
<td></td>
<td>\Phi_{vi} - \Phi_{ti}</td>
<td></td>
</tr>
<tr>
<td>$</td>
<td></td>
<td>\Phi_{si} - \Phi_{vi}</td>
<td></td>
</tr>
<tr>
<td>$</td>
<td></td>
<td>\Phi_{si} - \Phi_{vi}</td>
<td></td>
</tr>
<tr>
<td>$</td>
<td></td>
<td>\Phi_{fi} - \Phi_{vi}</td>
<td></td>
</tr>
<tr>
<td>$</td>
<td></td>
<td>\Phi_{fi} - \Phi_{vi}</td>
<td></td>
</tr>
<tr>
<td>$</td>
<td></td>
<td>\Phi_{fi} - \Phi_{si}</td>
<td></td>
</tr>
<tr>
<td>$</td>
<td></td>
<td>\Phi_{fi} - \Phi_{si}</td>
<td></td>
</tr>
</tbody>
</table>
Once the distances have been calculated, the various calculations for the future uncertainty can be made. The estimates for the uncertainty based on Eq. (4.6) are labeled with a subscript “\(L_p\)” in Table 8.25, while those based on the estimate from Eq. (4.8) are labeled with a subscript “\(L_e\)”, and those based on the uncertainty functions directly similar to (see Eq. (4.9)) are labeled with a subscript “\(u\)” as before. From this table and examining the number of misses, for all three models, the model uncertainty estimates based on the Lipschitz methods worked well, while the model uncertainty estimate limits based on the model uncertainty functions did not meet the risk requirements. As expected the estimates based on the Lipschitz approximate method, did quite well in ensuring it met the risk requirements and limiting the over-estimation as compared to the Lipschitz proof method.

<table>
<thead>
<tr>
<th></th>
<th>Fmincon w/ ( J_x )</th>
<th>GA</th>
<th>Fmincon w/ ( \dot{J}_x )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \Delta_m1(D_f)_Lp ) (V/s)</td>
<td>106.53</td>
<td>105.92</td>
<td>116.16</td>
</tr>
<tr>
<td>( \Delta_m2(D_f)_Lp ) (A/s)</td>
<td>169.4</td>
<td>168.07</td>
<td>198.81</td>
</tr>
<tr>
<td>( \Delta_m1(D_f)_Lc ) (V/s)</td>
<td>84.42</td>
<td>90.99</td>
<td>92.69</td>
</tr>
<tr>
<td>( \Delta_m2(D_f)_Lc ) (A/s)</td>
<td>106.27</td>
<td>157.17</td>
<td>123.1</td>
</tr>
<tr>
<td>( \Delta_m1(D_f)_u ) (V/s)</td>
<td>25.45</td>
<td>12.92</td>
<td>34.77</td>
</tr>
<tr>
<td>( \Delta_m2(D_f)_u ) (A/s)</td>
<td>50.52</td>
<td>32.03</td>
<td>48.91</td>
</tr>
<tr>
<td>( \Delta_m1(D_f) ) (V/s)</td>
<td>77.27</td>
<td>80.81</td>
<td>75.94</td>
</tr>
<tr>
<td>( \Delta_m2(D_f) ) (A/s)</td>
<td>136.42</td>
<td>162.47</td>
<td>135.11</td>
</tr>
<tr>
<td>( \hat{\Delta}_m1(D_f)_Lp ) misses</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>( \hat{\Delta}_m2(D_f)_Lp ) misses</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>( \hat{\Delta}_m1(D_f)_Lc ) misses</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>( \hat{\Delta}_m2(D_f)_Lc ) misses</td>
<td>3</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>( \hat{\Delta}_m1(D_f)_u ) misses</td>
<td>66</td>
<td>76</td>
<td>44</td>
</tr>
<tr>
<td>( \hat{\Delta}_m2(D_f)_u ) misses</td>
<td>25</td>
<td>76</td>
<td>42</td>
</tr>
</tbody>
</table>
8.4 Summary

In summary, the model uncertainty estimates based on the two Lipschitz methods performed well at predicting the uncertainty limits of the validation data set and the final data set, while the model uncertainty function estimate did not. Through other testing not presented here, what was found is that as long as the system is operated in a method that the model was designed for, the Lipschitz methods do perform well. If the system is operated in a way that the model was not designed for (e.g., discontinuous conduction mode where the inductor current goes to zero) then the estimates perform poorly. The further requirement is that the estimate of the Lipschitz constant and distance is done accurately. Future work should look at ensuring this has been accomplished, as several cases could be developed where the polynomial estimation may not be an accurate means of estimating the Lipschitz constant.
CHAPTER IX

CONTROL ALGORITHM VALIDATION

This chapter will present the results of the control uncertainty testing. It will be shown that the control uncertainty, $\delta_{c2}$, is large and it will be discussed why it was decided not to attempt fixing the issue now. Second it will be shown that $\delta_{c1}$ can be reduced by using the augmented control, and although this removes some uncertainty, not all uncertainty was removed. It will be shown that the control uncertainty limits as predicted by the Lipschitz methods are good predictions of the actual control uncertainty limits. Finally, some results of the new controllers’ ability to keep within an invariant set will be shown.

9.1 System and/or Parameter Identification (Augmented Controller Development)

The primary goal of this part of the controller verification and validation is determination as to whether the controller chosen can be used with the model form that was found in the previous chapter.

It was discussed in Section 7.2 that there was a delay of the duty cycle from the controller to the boost circuit. This delay is not well suited to be handled by a polynomial of the uncertainty, so at this time no action is done to reduce the uncertainty in $\delta_{c2}$. There is a possibility to use ideas from [83] for the delay, however future research will address the fundamental problem and replace the...
controller hardware with hardware that does not have the delay. It was shown in Section 6.5 that
without that delay the control uncertainty $\delta_{c2}$ was zero.

The other issue is the controller output voltage uncertainty, $\delta_{c1}$ is due to the approximation made
in Eq.(5.7). This estimation was made to put the plant into strict feedback form. The first task will
be to determine whether the existing uncertainty is acceptable via the baseline testing. Then if the
control uncertainty is unacceptable an augmented controller will be developed.

The results of the baseline testing for all four parameter sets is shown in Table 9.1 and demon-
strates the need for augmented controller for all four parameter sets (i.e., the requirements from
Section 5.5.2 of 50 V/s for $\Delta_{c1}$ and 150 A/s for $\Delta_{c2}$ will likely not be met). The uncertainty limit
for the baseline system is smaller than the other three systems. This difference is likely due to the
offset and gain modification applied to the duty cycle in the plant model to represent the dynamics
of the n-MOSFET (see Eq.(8.1)). The modification to the control based on the duty cycle changes
is the following

$$\dot{x}_2 = f_2 + g_2(d_T d_T g + d_{T_o}),$$

$$u = 1/(g_2 d_T g)(-f_2 - g_2 d_{T_o} + c\dot{c} - k_2 e_2 - \frac{g_1}{\beta_2^2} e_1).$$

What the above modification does not take into account is the slight modification to $\dot{V}_o$ from the
original differential equations (see Eq.(8.2)), to allow the control to remain in strict feedback form.
It will be seen later, that this difference is not an issue as the augmentation does perform better with
the GA, fmincon, and optimized derivative models over the baseline. Likely the influence is handled
by the augmentation.

The next two Tables 9.2 and 9.3 describe the augmented controllers for the fmincon and GA
based models. The augmented controllers for the parameter set from fmincon with $J_\dot{x}$ can be found
in Appendix I. The regression diagnostic Tables 9.4 and 9.5 demonstrate the augmented controllers
Table 9.1: Control Uncertainty Limit Calculations for Data Set $D_t$.

<table>
<thead>
<tr>
<th></th>
<th>Baseline</th>
<th>Fmincon</th>
<th>GA</th>
<th>Derivative Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\Delta c_1$ (V/s)</td>
<td>170.38</td>
<td>246.27</td>
<td>226.20</td>
<td>259.77</td>
</tr>
<tr>
<td>$\Delta c_2$ (A/s)</td>
<td>176.41</td>
<td>170.57</td>
<td>173.26</td>
<td>163.33</td>
</tr>
</tbody>
</table>

are good estimates of the uncertainty. Though without a delay, the control uncertainty cannot be reduced below 24 V/s as seen in the full plant and no delay example problem in Section 6.5.

Table 9.2: $\delta c_1$, Augmented Controller for Data Set $D_t$ for Model fmincon w/ $J_x$ cost function.

<table>
<thead>
<tr>
<th></th>
<th>Estimate</th>
<th>SE</th>
<th>tStat</th>
<th>pValue</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Intercept)</td>
<td>17.6</td>
<td>0.362</td>
<td>48.5</td>
<td>0</td>
</tr>
<tr>
<td>$I_i$</td>
<td>-553</td>
<td>2.37</td>
<td>-234</td>
<td>0</td>
</tr>
<tr>
<td>$V_i$</td>
<td>-9.67</td>
<td>0.0937</td>
<td>-103</td>
<td>0</td>
</tr>
<tr>
<td>$I_o$</td>
<td>-473</td>
<td>3.29</td>
<td>-144</td>
<td>0</td>
</tr>
<tr>
<td>$I_i : V_i$</td>
<td>262</td>
<td>0.651</td>
<td>402</td>
<td>0</td>
</tr>
<tr>
<td>$I_i : I_o$</td>
<td>-7.4e+03</td>
<td>11.9</td>
<td>-621</td>
<td>0</td>
</tr>
</tbody>
</table>

Control uncertainty estimates were created from the data set $D_v$ and are described in Tables 9.6, 9.7, 9.8, and 9.9. The control uncertainty polynomials found are similar with the exception that fmincon identified $\dot{I}_o$ as a term. They both had similar coefficients of determination, with a slight difference in the standard deviations.

9.2 Control Prediction and Hypothesis Testing

After generating the augmented controllers and developing the control uncertainties, the next step will be to evaluate the control uncertainties with the new data set $D_s$. First in that evaluation is calculation of the distances between data sets and that is displayed in Table 9.10. In the predictions, $||\Phi_{vi} - \Phi_{ti}||^{cu}$ will be used, and as one can see, that distance is similar to the distance that was
Table 9.3: $\delta_{c1}$, Augmented Controller for Data Set $D_t$ for Model GA.

<table>
<thead>
<tr>
<th>Estimate</th>
<th>SE</th>
<th>tStat</th>
<th>pValue</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Intercept)</td>
<td>36.3</td>
<td>0.482</td>
<td>75.4</td>
</tr>
<tr>
<td>$I_t$</td>
<td>-80.5</td>
<td>4.77</td>
<td>-16.9</td>
</tr>
<tr>
<td>$V_i$</td>
<td>-15.4</td>
<td>0.134</td>
<td>-115</td>
</tr>
<tr>
<td>$I_o$</td>
<td>-2.05e+03</td>
<td>18.7</td>
<td>-109</td>
</tr>
<tr>
<td>$\dot{I}_o$</td>
<td>-0.486</td>
<td>0.0052</td>
<td>-93.5</td>
</tr>
<tr>
<td>$I_t : V_i$</td>
<td>117</td>
<td>1.38</td>
<td>84.6</td>
</tr>
<tr>
<td>$I_t : I_o$</td>
<td>-6.73e+03</td>
<td>9.13</td>
<td>-738</td>
</tr>
<tr>
<td>$V_i : I_o$</td>
<td>457</td>
<td>5.33</td>
<td>85.8</td>
</tr>
</tbody>
</table>

Table 9.4: Regression Diagnostic Table for $\delta_{c1}$ (Augmented Controller) for Model fmincon w/ $J_x$ cost function.

<table>
<thead>
<tr>
<th>Std. Dev.</th>
<th>R-Squared</th>
<th>R-Squared Adj</th>
<th>pValue</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.7245</td>
<td>0.9961</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-64.9628</td>
<td>0.9956</td>
<td></td>
<td></td>
</tr>
<tr>
<td>373216</td>
<td>0.9961</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 9.5: Regression Diagnostic Table for $\delta_{c1}$ (Augmented Controller) for Model GA.

<table>
<thead>
<tr>
<th>Std. Dev.</th>
<th>R-Squared</th>
<th>R-Squared Adj</th>
<th>pValue</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.8520</td>
<td>0.9972</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-61.8823</td>
<td>0.9966</td>
<td></td>
<td></td>
</tr>
<tr>
<td>218851</td>
<td>0.9972</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 9.6: $\delta_{c1}$ Polynomial for Data Set $D_v$ for Model fmincon w/ $J_x$ cost function.

<table>
<thead>
<tr>
<th>Estimate</th>
<th>SE</th>
<th>tStat</th>
<th>pValue</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Intercept)</td>
<td>44</td>
<td>0.231</td>
<td>191</td>
</tr>
<tr>
<td>$I_t$</td>
<td>-424</td>
<td>3.08</td>
<td>-137</td>
</tr>
<tr>
<td>$I_o$</td>
<td>-667</td>
<td>4.4</td>
<td>-152</td>
</tr>
<tr>
<td>$\dot{I}_o$</td>
<td>-0.595</td>
<td>0.0049</td>
<td>-121</td>
</tr>
<tr>
<td>$I_t : I_o$</td>
<td>6.22e+03</td>
<td>38.4</td>
<td>162</td>
</tr>
</tbody>
</table>

Table 9.7: Regression Diagnostic table for $\delta_{c1}$ for Data Set $D_v$ for Model fmincon w/ $J_x$ cost function.

<table>
<thead>
<tr>
<th>Std. Dev.</th>
<th>R-Squared</th>
<th>R-Squared Adj</th>
<th>pValue</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1085</td>
<td>0.4645</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.9741</td>
<td>0.4422</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1078829</td>
<td>0.4644</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Table 9.8: $\delta_{c1}$ Polynomial for Data Set $D_v$ for Model GA.

<table>
<thead>
<tr>
<th></th>
<th>Estimate</th>
<th>SE</th>
<th>tStat</th>
<th>pValue</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Intercept)</td>
<td>42.4</td>
<td>0.204</td>
<td>208</td>
<td>0</td>
</tr>
<tr>
<td>$I_l$</td>
<td>-353</td>
<td>2.72</td>
<td>-130</td>
<td>0</td>
</tr>
<tr>
<td>$I_o$</td>
<td>-790</td>
<td>3.88</td>
<td>-203</td>
<td>0</td>
</tr>
<tr>
<td>$I_l : I_o$</td>
<td>6.11e+03</td>
<td>33.9</td>
<td>181</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 9.9: Regression Diagnostic Table for $\delta_{c1}$ for Model GA.

<table>
<thead>
<tr>
<th></th>
<th>Std. Dev.</th>
<th>R-Squared</th>
<th>R-Squared Adj</th>
<th>R-Squared Pred</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>3.6278</td>
<td>0.4450</td>
<td>0.4298</td>
<td>0.4450</td>
</tr>
</tbody>
</table>

predicted (i.e., $||\Phi_{vi} - \Phi_{ti}||^{cv} \approx ||\Phi_{svi} - \Phi_{ti}||^{cv}$). Further, by definition the validation data is strong validation data because both properties were met (i.e., $||\Phi_{vi} - \Phi_{ti}||^{cv} >> ||\Phi_{vi} - \Phi_{svi}||^{cv}$ is also true). The Fmincon w/ $J_x$ parameter set resulted in a distance $||\Phi_{vi} - \Phi_{ti}||^{cv} > ||\Phi_{vi} - \Phi_{svi}||^{cv}$ and not much greater as desired for a strong validation data set. Further work is necessary to understand the issue. The model of $\delta_{c1}$ are similar between the two parameter sets developed based on use of fmincon as seen in Tables 9.6 and 1.11. The augmented controllers are also very similar.

Table 9.10: Distance Calculation for Prediction of Data Set $D_s$.

<table>
<thead>
<tr>
<th></th>
<th>Baseline</th>
<th>Fmincon w/ $J_x$</th>
<th>GA</th>
<th>Fmincon w/ $J_x$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$</td>
<td></td>
<td>\Phi_{vi} - \Phi_{ti}</td>
<td></td>
<td>^{cv}$</td>
</tr>
<tr>
<td>$</td>
<td></td>
<td>\Phi_{svi} - \Phi_{ti}</td>
<td></td>
<td>^{cv}$</td>
</tr>
<tr>
<td>$</td>
<td></td>
<td>\Phi_{svi} - \Phi_{vi}</td>
<td></td>
<td>^{cv}$</td>
</tr>
</tbody>
</table>
The hypothesis testing indicates that all four parameter sets resulted in uncertainty estimate limits that were all confirmed via the hypothesis testing (see Table 9.11). Based on the Lipschitz method, the genetic algorithm predicted a slightly lower value for $\Delta c_1$, but actually had the largest measured control uncertainty limit of all three. The control uncertainty estimate limit based on the uncertainty functions also performed well for the genetic algorithm. Also note, even though the control uncertainty $\delta c_2$ is not accounted for (i.e., $\hat{\Delta}c_2(D_s)_L = \Delta c_2(D_v)$), the prediction of the limits based on other similar testing has predicted correctly as demonstrated here.

Table 9.11: Hypothesis Testing Results of Control Uncertainty for Data Set $D_s$.

<table>
<thead>
<tr>
<th></th>
<th>Baseline</th>
<th>Fmincon w/ $J_x$</th>
<th>GA</th>
<th>Fmincon w/ $J_{\bar{x}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\Delta c_1(D_t)$ (V/s)</td>
<td>170.38</td>
<td>246.27</td>
<td>226.20</td>
<td>259.77</td>
</tr>
<tr>
<td>$\Delta c_2(D_t)$ (A/s)</td>
<td>176.41</td>
<td>170.57</td>
<td>173.26</td>
<td>163.33</td>
</tr>
<tr>
<td>$\Delta c_1(D_v)$ (V/s)</td>
<td>60.12</td>
<td>43.02</td>
<td>39.73</td>
<td>45.70</td>
</tr>
<tr>
<td>$\Delta c_2(D_v)$ (A/s)</td>
<td>285.11</td>
<td>301.38</td>
<td>293.39</td>
<td>295.69</td>
</tr>
<tr>
<td>$\hat{\Delta}c_1(D_s)_L$ (V/s)</td>
<td>102.83</td>
<td>77.31</td>
<td>63.32</td>
<td>81.16</td>
</tr>
<tr>
<td>$\hat{\Delta}c_2(D_s)_L$ (A/s)</td>
<td>285.11</td>
<td>301.38</td>
<td>293.39</td>
<td>295.69</td>
</tr>
<tr>
<td>$\hat{\Delta}c_1(D_s)_u$ (V/s)</td>
<td>35.14</td>
<td>31.10</td>
<td>37.78</td>
<td>32.65</td>
</tr>
<tr>
<td>$\hat{\Delta}c_2(D_s)_u$ (A/s)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$\Delta c_1(D_s)$ (V/s)</td>
<td>131.84</td>
<td>43.68</td>
<td>46.71</td>
<td>36.11</td>
</tr>
<tr>
<td>$\Delta c_2(D_s)$ (A/s)</td>
<td>297.22</td>
<td>274.33</td>
<td>319.11</td>
<td>328.40</td>
</tr>
<tr>
<td>$\hat{\Delta}c_1(D_s)_L$ misses</td>
<td>4</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$\hat{\Delta}c_2(D_s)_L$ misses</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$\hat{\Delta}c_1(D_s)_u$ misses</td>
<td>76</td>
<td>8</td>
<td>1</td>
<td>7</td>
</tr>
<tr>
<td>$\hat{\Delta}c_2(D_s)_u$ misses</td>
<td>76</td>
<td>76</td>
<td>76</td>
<td>76</td>
</tr>
</tbody>
</table>

At this point, the augmented control algorithm has been validated and the projections estimates limits have also been validated. The data developed for $D_s$ is used to update the control uncertainty and then used for future prediction. The results of the new control uncertainty equations can be found in Tables 9.12 and 9.13 and again the only difference in these two polynomials is $\dot{I}_o$.

Also note, the equations based on data from $D_s$ used the same regressors as those equations based
on data from $D_v$ with the parameters estimates as the only difference. The regression diagnostics
demonstrate a similar behavior as before (see Table 9.14 and 9.15). The standard deviation difference
between the two models is more similar than before and both had improved coefficients of
determination.

Table 9.12: $\delta_{c1}$ Polynomial for Data Set $D_s$ for Model fmincon w/ $J_x$ Cost Function.

<table>
<thead>
<tr>
<th></th>
<th>Estimate</th>
<th>SE</th>
<th>tStat</th>
<th>pValue</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Intercept)</td>
<td>42.4</td>
<td>0.224</td>
<td>189</td>
<td>0</td>
</tr>
<tr>
<td>$I_l$</td>
<td>-401</td>
<td>3.11</td>
<td>-129</td>
<td>0</td>
</tr>
<tr>
<td>$I_o$</td>
<td>-615</td>
<td>4.32</td>
<td>-142</td>
<td>0</td>
</tr>
<tr>
<td>$\dot{I}_o$</td>
<td>-0.662</td>
<td>0.00512</td>
<td>-129</td>
<td>0</td>
</tr>
<tr>
<td>$I_l : I_o$</td>
<td>5.66e+03</td>
<td>38.6</td>
<td>147</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 9.13: $\delta_{c1}$ Polynomial for Data Set $D_s$ for Model GA.

<table>
<thead>
<tr>
<th></th>
<th>Estimate</th>
<th>SE</th>
<th>tStat</th>
<th>pValue</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Intercept)</td>
<td>44</td>
<td>0.229</td>
<td>192</td>
<td>0</td>
</tr>
<tr>
<td>$I_l$</td>
<td>-299</td>
<td>3.17</td>
<td>-94.1</td>
<td>0</td>
</tr>
<tr>
<td>$I_o$</td>
<td>-867</td>
<td>4.42</td>
<td>-196</td>
<td>0</td>
</tr>
<tr>
<td>$I_l : I_o$</td>
<td>5.62e+03</td>
<td>39.4</td>
<td>143</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 9.14: Regression Diagnostic Table for $\Delta_{c1}$ for Data Set $D_s$ for Model Fmincon w/ $J_x$ Cost Function.

<table>
<thead>
<tr>
<th></th>
<th>Std. Dev.</th>
<th>R-Squared</th>
<th>R-Squared Adj</th>
<th>R-Squared Pred</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean</td>
<td>1.9966</td>
<td>0.5129</td>
<td>0.4926</td>
<td>0.5128</td>
</tr>
<tr>
<td>PRESS</td>
<td>1142481</td>
<td>0.5129</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
9.3 Future Data Prediction and Future Data Analysis

As with the model validation work, the control uncertainties projections and the control uncertainty limits were used to predict the uncertainty in the future data. First starting with the distance calculations in Table 9.16 one can see that \( \|\Phi_{si} - \Phi_{ti}\|_{cs} > \|\Phi_{fi} - \Phi_{si}\|_{cs} \) for the three parameter sets and that \( \|\Phi_{si} - \Phi_{vi}\|_{cs} \approx \|\Phi_{fi} - \Phi_{si}\|_{cs} \), thus the Lipschitz approximate method is likely to be successful. Though it would not have been known for sure until the data had been taken.

<table>
<thead>
<tr>
<th></th>
<th>Fmincon w/ ( J_x )</th>
<th>GA</th>
<th>Fmincon w/ ( J_{\dot{x}} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( |\Phi_{si} - \Phi_{vi}|_{cs} )</td>
<td>8.20</td>
<td>3.52</td>
<td>10.92</td>
</tr>
<tr>
<td>( |\Phi_{fi} - \Phi_{vi}|_{cs} )</td>
<td>2.63</td>
<td>2.54</td>
<td>2.85</td>
</tr>
<tr>
<td>( |\Phi_{fi} - \Phi_{si}|_{cs} )</td>
<td>10.14</td>
<td>5.73</td>
<td>8.82</td>
</tr>
<tr>
<td>( |\Phi_{si} - \Phi_{ti}|_{cs} )</td>
<td>16.70</td>
<td>14.07</td>
<td>15.48</td>
</tr>
</tbody>
</table>

The results of the hypothesis testing does show that the Lipschitz proof and approximate method passed (i.e., the number of misses was less than or equal to seven). The Lipschitz proof estimates as expected were slightly higher than the approximate methods as expected. And all three parameter sets resulted in similar estimation of the control uncertainty limits with the genetic algorithm estimate indicating slightly better performance. However, all three were within 1 V/s difference with the measured uncertainty.
9.4 Summary

In this chapter it has been demonstrated that the control uncertainty limit estimates have been validated and one could predict other data. However, using the predicted uncertainties of the model uncertainty found in the Model Validation chapter, Chapter VIII, and the control uncertainties found in this chapter, the accuracy requirements for the total uncertainty limit, \( \Delta_1 \) was met. However, the requirement for the total uncertainty limit, \( \Delta_2 \), was not. Figure 9.1 shows the theoretical invariant set for this problem based on the uncertainties found for the fmincon parameter set. The red area is composed of points along a potential state trajectory (i.e., region of attraction) that will converge into the green area (i.e., invariant set). The green area is the uniform ultimate bound area or what has been called the invariant set. This map was made by varying the output voltage from 4 V to 9 V and inductor current from 0 A to 1 A, calculating a current command from the augmented control
law, calculating the Lyapunov function evaluation from the error estimates and ensuring that they were less than a bound (see Eq. (5.14)). The code is included in Appendix G.

As expected, the invariant set was much smaller than the theoretical invariant set as seen in Figs. 9.2, 9.3, 9.4, and 9.5. Figure 9.2 displays the invariant set for the original data set $D_v$. It does not use an augmented controller, and does not use the duty cycle modification, nor does it include the second order diode modification in the controller. As can be seen, this controller performed well, though the region is not centered on zero error. The other three figures do show the error centered on zero, but the overall area is similar. It is shown that the high level requirement of ±200 mV tracking
was accomplished via demonstration but was not shown through Lyapunov stability analysis. Note, the red ‘+’ symbol is the state of the system at the time immediately before the change in loading (i.e., time of 0.09 s). The green ‘o’ symbol is the state of the system at a time of 0.11 s, after the load transition. The blue lines are the states of the system between those two times.

Figure 9.2: Original data, invariant set.

Figure 9.6 was created based on the simulation data set $D_s$ using the fmincon parameter set and augmented controller. The invariant set is about half the length in the x-axis (i.e., $e_1$) and similar in the y-axis compared to the experimental data (see Fig. 9.3). For parameter set fmincon, the total uncertainty $\Delta_1$ is equal to 144.5 V/s with 84.4 V/s as a result of the model uncertainty $\Delta_{m1}$ and 60.08 as a result of the control uncertainty $\Delta_{c1}$. A little more than half of the uncertainty is
Figure 9.3: Invariant set of the Fmincon Parameter Set.
Figure 9.4: Invariant set of the GA parameter set.
Figure 9.5: Invariant set of the Derivative Output parameter set.
from the model and thus the simulated invariant set should be less accurate in that dimension as it only includes the controller uncertainty. While the total uncertainty from $\Delta_2$ is 380.6 A/s with $\Delta_{m2}$ equal to 106.27 and $\Delta_{c2}$ equal to 274.33. Thus the total uncertainty is not as dominated by the model uncertainty in that dimension and the simulated invariant set in that dimension is more similar to the experimental data. Further, Fig. 9.7 demonstrates the improvements that could occur if the delay is removed from the control. Again, understanding the model uncertainties would help understand the benefits. Future work could include examining means of estimating the effects of the model uncertainty on the invariant set of the simulation run and not rely on the conservative approach of Lyapunov stability analysis.

Figure 9.6: Simulated invariant set of the Fmincon parameter set.
Figure 9.7: Simulated invariant set for Example Problem 5.
CHAPTER X

FUTURE RESEARCH AND CONCLUSIONS

10.1 Future Research

Below are a number of future research ideas that would complement the research presented in this dissertation.

To improve performance for the dc-dc boost circuit it is strongly recommended to use different control hardware and/or a different controller operating system to increase the control rate and remove the time delay $T_d$. It was demonstrated in the example problem in Section 6.5 that without the delay, the control uncertainty $\delta_{c2}$ was zero when the delay was removed. Obviously, not all the delay can be removed, but a substantial piece of it can be.

In this work, the validation test vectors were picked before the uncertainty calculations were done. A potentially better solution would be to combine a Halton sequence for specifying the operating condition (e.g., input voltage) and use ideas from optimal control to develop time varying signals for the duty cycle and resistor switching characteristic that maximizes the model and/or control uncertainty while using the known physics as constraints.

In this work, the beta modification was only shown to change the characteristics of the invariant set via a Lyapunov stability analysis. Are similar types of effects seen with experimentally or simulation tests? Further work could be done to test those theories.
Another research idea is to examine other means of more accurately estimating the uncertainty functions and/or the Lipschitz constant and distance. As was seen, the estimation of the maximum uncertainty from the uncertainty equations directly tended to under-estimate the uncertainty limits. Techniques such as principal component regression or partial least squares were applied to the problem but offered little advantage, are there other techniques? Techniques considered should be differentiable so that a Lipschitz constant and distance can be easily calculated.

The goal in this dissertation was to estimate the uncertainty of the differential equation $\delta_i$ and in many robust control applications that is useful. However, in other applications there is a desire to perform model validation only and estimate the uncertainty of the outputs. Is there a means of applying similar ideas presented here to that problem? Specifically, is there a means to calculate a dynamic uncertainty and project that uncertainty to predict future data?

It was found in the last chapter that the Lyapunov stability analysis was a very conservative means of estimating the invariant set, and using only the control uncertainty produced an invariant set smaller than the actual. A means to include the model uncertainty into a calculation or simulation to determine the invariant set would be very beneficial for future work.

In this work there is an assumption that the Lipschitz constant is similar across the data sets. Are there means to remove that restriction thus make it a more powerful approach to prediction?

10.2 Conclusions

Throughout this dissertation a presentation of a method and application of that method was performed to demonstrate a statistically based means of model and control algorithm validation. Through the dc-dc boost circuit problem, a demonstration that the total uncertainty can be decomposed into a model uncertainty analysis that does require experimental data and a control uncertainty analysis that only requires simulations. The control uncertainty calculation could be made
more robust by more extensive simulation studies assuming those tests are relatively less expensive. The hypothesis tests demonstrated the confirmation of the predicted uncertainty limits for the validation data and the future data set within the risk requirements specified for both the model and the control. In the model validation case, knowledge of the model uncertainty limit allows the user of the model (i.e., the control algorithm developer) to understand the uncertainty in the model and make decisions on whether the control algorithm developed will meet its requirements without having to perform separate experimental testing.

The process developed reduced the amount of testing through two means. First, it made use of design of experiments to limit the number of tests for any given data set. For example, the tuning experiments used a D-optimal design for specifying the boundary conditions. The validation tests used a Halton sequence to estimate the population behavior. These quasi-random sequences have been shown to require fewer tests than pseudo-random sequences [48] and [49]. The second means of reducing the amount of testing was through use of the validated models. The validation test data, $D_v$ was generated once for model validation and the future data set, $D_f$ was generated once. Thus, the control testing was not the typical trial and error. The control testing consisted of one set of tests because the confidence in the model was high. The model validation was also only performed once because the screening and tuning was thorough. The screening and system identification involved more than one set test runs to identify the issues in data acquisition and model form, but these only required the baseline control development and could possibly had been done without the baseline control development.

By using the statistical techniques of probability of acceptance and relying on tolerance interval calculations that were distribution free, assumptions about the error distributions were not needed. Thus, any distribution of model form error could be evaluated using the methods presented here.
Using this process for control algorithm validation gives the control algorithm developer freedom
to use forms of a control model that could not have been evaluated before.

The one problem not successfully addressed was proof of the dc-dc boost converter control per-
formance. The predictions and actual control uncertainty limits were larger than the requirements.
However, it was demonstrated that improvements were made over the baseline control algorithm by
following these steps. Further, understanding of the limitations of the current system were identified
(i.e., delay in the controller and control loop rate). This process showed measurable improvements
can be demonstrated in simulation before different control hardware is purchased. Having the con-
fidence through this analysis to show the improvements versus the costs can make decisions consid-
erably easier for the stakeholders (i.e., how much money would it cost to use a controller without a
delay vs. how much of an improvement would likely be seen).

The other two key contributions from this dissertation made in the field of model and control
algorithm validation include the measurement of strong validation data and projection of the model
and control uncertainty limits into the application space. The strong validation data as discussed
in Section 4.1.2 can help direct the planning the experimental validation data. In the situation pre-
sent here, the choice was to make the validation similar to the application data and thus only an
evaluation was done as to whether the validation data was strong or not. Future research should ex-
plode the possibility of designing the validation experiment to ensure strong validation data through
maximization of the distance from the validation data set to the tuning data set.

Finally the last contribution is that of projection of the model and control uncertainty limits into
the validation space and into the application space. In Section 4.1 use of the Lipschitz condition was
utilized to show mathematically the projection of the uncertainty limits. It was then demonstrated
in Chapters VIII and IX that those projections met the risk requirements. Obviously, there are
situations where the Lipschitz condition will not be satisfied in the application space. Also there is
no means to predict whether the Lipschitz condition will be satisfied. Thus model validation does not preclude additional testing, the goal is to reduce the amount of testing. However, it is believed that this assumption is the first step in the process of better model uncertainty predictions and could be made stronger in the future (i.e., see the recommendation for future research).


[16] Verification, Validation, and Accreditation of Army Models and Simulations, United States Army Std. Pamphlet 5-11, September 1999.


APPENDIX A

TEST VECTORS

In Table A.1 and A.2, $\dot{I}_o$ was not directly commanded in the experiment. Instead, if $\dot{I}_o$ is greater than zero then a transition from one resistor loading the circuit to two resistors loading the circuit was performed. The reverse occurred if $\dot{I}_o$ was less than 0.
Table A.1: Test Vector for the Tuning Data Set $D_t$.

<table>
<thead>
<tr>
<th>Run</th>
<th>$A:V_{oref}$</th>
<th>$B:V_i$</th>
<th>$C: \dot{i}_o$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>7</td>
<td>3.5</td>
<td>-40</td>
</tr>
<tr>
<td>2</td>
<td>9.5</td>
<td>4</td>
<td>-40</td>
</tr>
<tr>
<td>3</td>
<td>9.5</td>
<td>4</td>
<td>40</td>
</tr>
<tr>
<td>4</td>
<td>9.5</td>
<td>3</td>
<td>-40</td>
</tr>
<tr>
<td>5</td>
<td>4.5</td>
<td>3</td>
<td>-40</td>
</tr>
<tr>
<td>6</td>
<td>4.5</td>
<td>3.5</td>
<td>40</td>
</tr>
<tr>
<td>7</td>
<td>4.5</td>
<td>4</td>
<td>40</td>
</tr>
<tr>
<td>8</td>
<td>7</td>
<td>4</td>
<td>40</td>
</tr>
<tr>
<td>9</td>
<td>9.5</td>
<td>3</td>
<td>40</td>
</tr>
<tr>
<td>10</td>
<td>7</td>
<td>3.5</td>
<td>40</td>
</tr>
<tr>
<td>11</td>
<td>9.5</td>
<td>3</td>
<td>40</td>
</tr>
<tr>
<td>12</td>
<td>4.5</td>
<td>4</td>
<td>-40</td>
</tr>
<tr>
<td>13</td>
<td>7</td>
<td>3.5</td>
<td>40</td>
</tr>
<tr>
<td>14</td>
<td>9.5</td>
<td>3</td>
<td>-40</td>
</tr>
<tr>
<td>15</td>
<td>9.5</td>
<td>3.5</td>
<td>-40</td>
</tr>
<tr>
<td>16</td>
<td>7</td>
<td>3.5</td>
<td>40</td>
</tr>
<tr>
<td>17</td>
<td>4.5</td>
<td>4</td>
<td>-40</td>
</tr>
<tr>
<td>18</td>
<td>7</td>
<td>3</td>
<td>40</td>
</tr>
<tr>
<td>19</td>
<td>7</td>
<td>3.5</td>
<td>-40</td>
</tr>
<tr>
<td>20</td>
<td>4.5</td>
<td>3</td>
<td>-40</td>
</tr>
<tr>
<td>21</td>
<td>9.5</td>
<td>4</td>
<td>-40</td>
</tr>
<tr>
<td>22</td>
<td>7.102158</td>
<td>3</td>
<td>-40</td>
</tr>
<tr>
<td>23</td>
<td>4.5</td>
<td>3</td>
<td>40</td>
</tr>
<tr>
<td>24</td>
<td>4.5</td>
<td>4</td>
<td>40</td>
</tr>
<tr>
<td>25</td>
<td>4.5</td>
<td>3</td>
<td>40</td>
</tr>
<tr>
<td>26</td>
<td>9.5</td>
<td>3.5</td>
<td>40</td>
</tr>
<tr>
<td>27</td>
<td>4.5</td>
<td>3</td>
<td>-40</td>
</tr>
<tr>
<td>28</td>
<td>7.116613</td>
<td>4</td>
<td>-40</td>
</tr>
<tr>
<td>29</td>
<td>7</td>
<td>3.5</td>
<td>-40</td>
</tr>
<tr>
<td>30</td>
<td>9.5</td>
<td>3</td>
<td>-40</td>
</tr>
<tr>
<td>31</td>
<td>9.5</td>
<td>4</td>
<td>40</td>
</tr>
<tr>
<td>32</td>
<td>4.5</td>
<td>4</td>
<td>-40</td>
</tr>
</tbody>
</table>
Table A.2: Test Vector for the Data Sets $D_v$, $D_{sv}$, $D_s$, and $D_f$.

<table>
<thead>
<tr>
<th>Run</th>
<th>$A: V_{oref}$</th>
<th>$B: V_i$</th>
<th>$C: \dot{I}_o$</th>
<th>Run</th>
<th>$A: V_{oref}$</th>
<th>$B: V_i$</th>
<th>$C: \dot{I}_o$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>5.5378</td>
<td>3.3177</td>
<td>40</td>
<td>39</td>
<td>6.1756</td>
<td>3.8638</td>
<td>-40</td>
</tr>
<tr>
<td>3</td>
<td>6.6608</td>
<td>3.8266</td>
<td>40</td>
<td>41</td>
<td>5.4017</td>
<td>3.348</td>
<td>40</td>
</tr>
<tr>
<td>4</td>
<td>8.7897</td>
<td>3.0694</td>
<td>40</td>
<td>42</td>
<td>7.5306</td>
<td>3.9488</td>
<td>-40</td>
</tr>
<tr>
<td>5</td>
<td>4.5466</td>
<td>3.422</td>
<td>-40</td>
<td>43</td>
<td>6.9349</td>
<td>3.1857</td>
<td>40</td>
</tr>
<tr>
<td>6</td>
<td>7.7302</td>
<td>3.8952</td>
<td>40</td>
<td>44</td>
<td>8.3607</td>
<td>3.4779</td>
<td>40</td>
</tr>
<tr>
<td>7</td>
<td>6.0798</td>
<td>3.1484</td>
<td>-40</td>
<td>45</td>
<td>4.8206</td>
<td>3.6836</td>
<td>-40</td>
</tr>
<tr>
<td>8</td>
<td>9.322</td>
<td>3.5517</td>
<td>40</td>
<td>46</td>
<td>8.0628</td>
<td>3.2749</td>
<td>40</td>
</tr>
<tr>
<td>9</td>
<td>5.3571</td>
<td>3.7574</td>
<td>-40</td>
<td>47</td>
<td>5.973</td>
<td>3.637</td>
<td>-40</td>
</tr>
<tr>
<td>10</td>
<td>7.5483</td>
<td>3.2334</td>
<td>-40</td>
<td>48</td>
<td>9.1273</td>
<td>3.8043</td>
<td>40</td>
</tr>
<tr>
<td>11</td>
<td>6.8647</td>
<td>3.5997</td>
<td>40</td>
<td>49</td>
<td>5.6312</td>
<td>3.0225</td>
<td>-40</td>
</tr>
<tr>
<td>13</td>
<td>4.9409</td>
<td>3.0962</td>
<td>40</td>
<td>51</td>
<td>6.4942</td>
<td>3.9715</td>
<td>40</td>
</tr>
<tr>
<td>14</td>
<td>8.0073</td>
<td>3.3345</td>
<td>-40</td>
<td>52</td>
<td>8.6231</td>
<td>3.1403</td>
<td>-40</td>
</tr>
<tr>
<td>15</td>
<td>5.9907</td>
<td>3.9353</td>
<td>-40</td>
<td>53</td>
<td>4.7462</td>
<td>3.5093</td>
<td>40</td>
</tr>
<tr>
<td>16</td>
<td>9.0571</td>
<td>3.2151</td>
<td>40</td>
<td>54</td>
<td>7.8126</td>
<td>3.7274</td>
<td>40</td>
</tr>
<tr>
<td>17</td>
<td>5.7368</td>
<td>3.4607</td>
<td>-40</td>
<td>55</td>
<td>6.3673</td>
<td>3.298</td>
<td>-40</td>
</tr>
<tr>
<td>18</td>
<td>7.1333</td>
<td>3.6747</td>
<td>40</td>
<td>56</td>
<td>9.3897</td>
<td>3.5902</td>
<td>40</td>
</tr>
<tr>
<td>19</td>
<td>6.4497</td>
<td>3.2618</td>
<td>40</td>
<td>57</td>
<td>5.1905</td>
<td>3.8453</td>
<td>-40</td>
</tr>
<tr>
<td>20</td>
<td>8.6445</td>
<td>3.6651</td>
<td>-40</td>
<td>58</td>
<td>7.4073</td>
<td>3.0506</td>
<td>40</td>
</tr>
<tr>
<td>22</td>
<td>7.9218</td>
<td>3.0089</td>
<td>-40</td>
<td>60</td>
<td>8.5596</td>
<td>3.9176</td>
<td>-40</td>
</tr>
<tr>
<td>23</td>
<td>6.2714</td>
<td>3.3752</td>
<td>40</td>
<td>61</td>
<td>5.0123</td>
<td>3.1851</td>
<td>40</td>
</tr>
<tr>
<td>25</td>
<td>5.2119</td>
<td>3.1351</td>
<td>-40</td>
<td>63</td>
<td>5.8424</td>
<td>3.7516</td>
<td>40</td>
</tr>
<tr>
<td>26</td>
<td>7.3408</td>
<td>3.5014</td>
<td>40</td>
<td>64</td>
<td>8.9088</td>
<td>3.2537</td>
<td>-40</td>
</tr>
<tr>
<td>27</td>
<td>6.833</td>
<td>3.714</td>
<td>-40</td>
<td>65</td>
<td>5.5885</td>
<td>3.6181</td>
<td>-40</td>
</tr>
<tr>
<td>28</td>
<td>8.4638</td>
<td>3.3271</td>
<td>40</td>
<td>66</td>
<td>7.2047</td>
<td>3.8732</td>
<td>40</td>
</tr>
<tr>
<td>29</td>
<td>5.0849</td>
<td>3.5679</td>
<td>40</td>
<td>67</td>
<td>6.6383</td>
<td>3.0776</td>
<td>-40</td>
</tr>
<tr>
<td>30</td>
<td>8.0946</td>
<td>3.8354</td>
<td>-40</td>
<td>68</td>
<td>8.7379</td>
<td>3.3698</td>
<td>40</td>
</tr>
<tr>
<td>31</td>
<td>5.7703</td>
<td>3.0398</td>
<td>40</td>
<td>69</td>
<td>4.5386</td>
<td>3.9583</td>
<td>40</td>
</tr>
<tr>
<td>33</td>
<td>5.4871</td>
<td>3.9081</td>
<td>40</td>
<td>71</td>
<td>6.1021</td>
<td>3.4507</td>
<td>40</td>
</tr>
<tr>
<td>34</td>
<td>7.2645</td>
<td>3.1619</td>
<td>40</td>
<td>72</td>
<td>9.2857</td>
<td>3.6916</td>
<td>-40</td>
</tr>
<tr>
<td>35</td>
<td>6.5369</td>
<td>3.5241</td>
<td>-40</td>
<td>73</td>
<td>5.2915</td>
<td>3.2924</td>
<td>40</td>
</tr>
<tr>
<td>36</td>
<td>8.8416</td>
<td>3.7664</td>
<td>40</td>
<td>74</td>
<td>7.5962</td>
<td>3.6463</td>
<td>-40</td>
</tr>
<tr>
<td>37</td>
<td>4.6424</td>
<td>3.2438</td>
<td>-40</td>
<td>75</td>
<td>6.9126</td>
<td>3.789</td>
<td>-40</td>
</tr>
<tr>
<td>38</td>
<td>7.6795</td>
<td>3.61</td>
<td>40</td>
<td>76</td>
<td>8.2944</td>
<td>3.0317</td>
<td>40</td>
</tr>
</tbody>
</table>
APPENDIX B

TOLERANCE INTERVAL CALCULATIONS

Tolerance intervals are used extensively in this work and because of that, an analysis was sought to understand how well various calculations of tolerance intervals performed. The calculations used here can be broadly grouped into three areas: calculations based on the normal assumption, calculations based on order statistics, and calculations based on Gaussian mixture modeling. A simulation test was devised to compare the various methods against a number of different distributions to better understand the strengths and weaknesses of each. The idea for this test came from the paper [84] where methods for calculating the confidence interval for the variance in a random sample were performed. The majority of distributions suggested in [84] will be used to evaluate methods for calculating the upper limit of the tolerance interval.

B.1 Methods for Calculating the Tolerance Interval

The first method reported on was described with Eq.(2.6) in Section 2.4.5 and assumes normality of the data. The goal was to examine how well this method performs with low number of samples and how well other data with non-normal distributions affect the calculation.

The next three methods examined fall within the general category of ordered statistics. The first method is described in Section 2.4.5 and Eqs.(2.7) and will be called the baseline ordered statistic approach. As an alternative to this approach, the probability $P(F(U) > p_1)$ was maximized by
selecting the maximum value of the data set and setting that as the upper limit. This method will be called the max point method. The final method based on ordered statistics is one where the expected value of alpha is more closely aligned with the desired value of alpha. For example, if one uses the max points method to determine the tolerance interval, then based on the number of points, the expected alpha is seen on the y-axis of Fig. B.1. To exceed an alpha level of 0.8, 32 points are required. Obviously, the more points used for calculating the upper limit of the tolerance interval, the larger alpha becomes. Figure B.2 demonstrates the value of alpha using the baseline method. In that figure, there are several instances where the confidence level exceeds the desired level of alpha (e.g., greater than 32 samples and less than 58 samples). To rectify this problem, an algorithm was developed that in this case, if $N$ was between 32 and 58 inclusive, then the data set would be randomly downsampling and only 32 points would be used in a max points tolerance interval calculation, thus ensuring one finds a confidence level, alpha, meeting the desired confidence level. A figure of the theoretical results of this downsampling method can be seen in Fig. B.3. In Figs. B.1 - B.3, alpha was set to 0.80. To ensure these methods worked for other values of alpha, a value of 0.95 was used (see Figs. B.4 - B.6).

Figure B.1: Non-Parametric Tolerance Interval (One-Sided) for the Max Points Method ($\alpha = 0.8$).
The last calculation implemented, the maximum method, was based on a combination of the methods above. Figure B.4 depicts the minimum number of points needed to meet levels of alpha for a given sample size using ordered statistics. Unfortunately for this work, to obtain a 95% confidence level (i.e., $1 - \alpha$) 59 points are required in the max points tolerance interval calculation and at times fewer data points are available (e.g., in the tuning phases only 32 data points are used). As a means to solve this problem, the tolerance levels will be based on a maximum of the downsampling
Figure B.4: Non-Parametric Tolerance Interval (One-Sided) for Max Points Method ($\alpha = 0.95$).

Figure B.5: Non-Parametric Tolerance Interval (One-Sided) for the Baseline Method ($\alpha = 0.95$).

method of the ordered statistics and the method based on the normality assumption if the ordered statistics calculation is pre-determined to have a lower value of alpha given the number of data points available.

Let $U_1$ be equal to the upper limit of the tolerance interval as calculated by downsampling method using ordered statistics. Let $U_2$ be equal to the upper tolerance interval as calculated by the
method based on the normality assumption. Then

\[ if(1 - F_\beta(p_1, N, 1) < \alpha) \]

\[ U = \max(U_1, U_2), \]

\[ else \]

\[ U = U_1, \]

\[ end \]

where \( F_\beta \) is the cumulative beta distribution, and \( N \) is the number of points available for the tolerance interval calculation. It will be proven in Section B.4 that

\[ P(F(\max(U_1, U_2)) > p_1) \geq \max(P(F(U_1) > p_1), P(F(U_2) > p_2)). \]  

(B.1)

Thus this new method will be an upper bound on either of the other two methods and for low data point counts will ensure maximum confidence. Although there is a chance of overestimation, that is preferred over underestimation.

Finally, although the results are not presented here, a method based on Gaussian mixture modeling was attempted. This method consisted of identification of a two component Gaussian mixture...
model using MATLAB’s function `gmdistributions.fit`. A random sample of 50,000 points is then constructed from the identified distribution to calculate a tolerance interval. A number of issues were found. First, for low sample sizes (e.g., 8) the fitting of the distribution tended to be poor and thus the estimate for the tolerance interval and other statistics was poor. Second, more work was necessary to calculate a tolerance interval based on the fitted population. This method, primarily because of the fitting problem was abandoned.

### B.2 Simulation Study Description

A broad range of distributions and varying parameters were tested to ensure that the methods studied will likely result in a good estimation of the tolerance interval, as it can be difficult to predict the residual distribution of the models and experimental data. The thirteen studied distributions are listed in Table B.1.

In the Table B.1, \( B(a, b) \) is the beta function as described in [60], and \( I_{(0,1)}(x) \) is a function such that only values of \( x \) between 0 and 1 have non-zero values [81]. All but the last two distributions are natively supported by MATLAB [81]. A sample of the contaminated distribution was generated with 500,000 points and used to calculate the properties of the distribution. For those not familiar with the contaminated normal distribution, it is characterized as a standard normal distribution with a probability of \( \lambda_p \) for each point taken from a normal distribution with mean of 0 and variance \( \sigma^2 \).

For the study, each distribution detailed in Table B.1 and each method mentioned earlier was tested with sample sizes of 8, 16, 32, 50, and 76. For each evaluation, 5000 replications were executed. For each replication, a coverage factor was determined by calculating \( F(U) \) for each distribution and determining if that exceeded \( p_1 \). The number of times the coverage factor exceeded \( p_1 \) was tracked and divided by 5000 to arrive at a value for alpha of the test.
Table B.1: List of Distributions used in Tolerance Interval study.

<table>
<thead>
<tr>
<th>Distribution Name</th>
<th>Parameters</th>
<th>Notation</th>
<th>PDF</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normal</td>
<td>$\mu = 0, \sigma = 1$</td>
<td>N(0,1)</td>
<td>$\frac{1}{\sqrt{2\pi\sigma^2}}e^{-\frac{(x-\mu)^2}{2\sigma^2}}$</td>
</tr>
<tr>
<td>Normal</td>
<td>$\mu = 80, \sigma = 10$</td>
<td>N(80,10)</td>
<td>$\frac{1}{\sqrt{2\pi\sigma^2}}e^{-\frac{(x-\mu)^2}{2\sigma^2}}$</td>
</tr>
<tr>
<td>Student T</td>
<td>$\mu = 0, \sigma = 1, \nu = 10$</td>
<td>t(0,1,10)</td>
<td>$\frac{\Gamma\left(\frac{\nu+1}{2}\right)}{\sqrt{\nu\pi}\Gamma\left(\frac{\nu}{2}\right)}\left[\frac{\nu+\left(\frac{x-\mu}{\sigma}\right)^2}{\nu}\right]^{-\frac{(\nu+1)}{2}}$</td>
</tr>
<tr>
<td>Student T</td>
<td>$\mu = 0, \sigma = 1, \nu = 5$</td>
<td>t(0,1,5)</td>
<td>$\frac{\Gamma\left(\frac{\nu+1}{2}\right)}{\sqrt{\nu\pi}\Gamma\left(\frac{\nu}{2}\right)}\left[\frac{\nu+\left(\frac{x-\mu}{\sigma}\right)^2}{\nu}\right]^{-\frac{(\nu+1)}{2}}$</td>
</tr>
<tr>
<td>Uniform</td>
<td>$a = 0, b = 1$</td>
<td>U(0,1)</td>
<td>$\frac{1}{b-a}I(0,1)$</td>
</tr>
<tr>
<td>Beta</td>
<td>$a = 3, b = 3$</td>
<td>B(3,3)</td>
<td>$\frac{1}{B(a,b)}x^{a-1}(1-x)^{b-1}I_{(0,1)}(x)$</td>
</tr>
<tr>
<td>Gamma</td>
<td>$a = 2, b = 3$</td>
<td>G(2,3)</td>
<td>$\frac{1}{B(a,b)}x^{a-1}e^{-\frac{x}{b}}$</td>
</tr>
<tr>
<td>Gamma</td>
<td>$a = 5, b = 6$</td>
<td>G(5,6)</td>
<td>$\frac{1}{B(a,b)}x^{a-1}e^{-\frac{x}{b}}$</td>
</tr>
<tr>
<td>Exponential</td>
<td>$\mu = 1$</td>
<td>E(1)</td>
<td>$\frac{1}{\mu}e^{-\frac{x}{\mu}}$</td>
</tr>
<tr>
<td>Beta</td>
<td>$a = 0.4, b = 0.7$</td>
<td>B(0.4,0.7)</td>
<td>$\frac{1}{B(a,b)}x^{a-1}(1-x)^{b-1}I_{(0,1)}(x)$</td>
</tr>
<tr>
<td>Beta</td>
<td>$a = 8, b = 1$</td>
<td>B(8,1)</td>
<td>$\frac{1}{B(a,b)}x^{a-1}(1-x)^{b-1}I_{(0,1)}(x)$</td>
</tr>
<tr>
<td>Contaminated Normal</td>
<td>$\lambda_p = 0.8, \sigma = 3$</td>
<td>CN(0.8,3)</td>
<td>See Text</td>
</tr>
<tr>
<td>Contaminated Normal</td>
<td>$\lambda_p = 0.9, \sigma = 3$</td>
<td>CN(0.9,3)</td>
<td>See Text</td>
</tr>
</tbody>
</table>

All of those tests were repeated for different values of desired alpha (e.g., 0.8 and 0.95). An Alpha value of 0.95 was used due to the risk requirements (see Section 5.5.2). The addition of the testing at value of alpha equal to 0.8 was done to demonstrate some of the over-prediction and highlight the differences between the methods.

### B.3 Simulation Study Results

The results of the simulation study are captured in Tables B.2 and B.3. Some items to note in the results for the method based on the normal assumptions. The method obviously performed well for the normal distributions and the student t-distributions as one would expect. The method underestimated the tolerance interval for both gamma, the exponential and one of the beta distributions, $B(0.4, 0.7)$. The normal assumption method overestimated the beta distribution, $B(8, 1)$, and the...
uniform distribution slightly. This method had problems with non-symmetric distributions as might be expected.

From these tables, one can see that the various ordered statistics methods had difficulties with the low sample numbers (as predicted) but performed well once the minimum number of samples was exceeded. The baseline and max points methods did over-predict at times as was expected, with the baseline method not over-predicting as much. The downsampling method on the other hand performed very well at not over-predicting alpha.

Finally, the maximum method performed very well. Examining Table B.2 one can see that it only over-predicted on the beta distribution (B(8,1)), the exponential distribution, and the uniform distribution as the normal assumption had over-predicted with these distribution. It did slightly better than the normal method at not under-predicting with the low counts of the exponential distribution and the two gamma distributions. It can be seen that Eq.(B.1) did hold throughout. In Table B.3 similar results can be seen with the exception of the beta distribution (B(8,1)) where over-prediction occurred at samples of 32 and 50. Also note, at sample count of 50, the normal and student-t distributions were over-predicted by 2-3% due to the combination of the other two prediction intervals. But this slight over-prediction is more than worth it, due to the desire to find a maximum value and the payoff received by not under-predicting as much on some of the other intervals.

Throughout the dissertation, when a tolerance interval level was reported it used the maximum method as reported here.

The code used to execute the simulation can be found in Appendix B.8.
<table>
<thead>
<tr>
<th>N</th>
<th>N(0,1)</th>
<th>N(0,10)</th>
<th>N(1,0)</th>
<th>N(1,10)</th>
<th>N(0,5)</th>
<th>N(0,10)</th>
<th>G(2,3)</th>
<th>G(5,6)</th>
<th>E(1)</th>
<th>B(6,4,0.7)</th>
<th>B(8,1)</th>
<th>CN(0,8,3)</th>
<th>CN(0,9,3)</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>0.356</td>
<td>0.355</td>
<td>0.356</td>
<td>0.355</td>
<td>0.356</td>
<td>0.355</td>
<td>0.356</td>
<td>0.355</td>
<td>0.356</td>
<td>0.356</td>
<td>0.355</td>
<td>0.356</td>
<td>0.356</td>
</tr>
<tr>
<td>16</td>
<td>0.556</td>
<td>0.556</td>
<td>0.556</td>
<td>0.556</td>
<td>0.556</td>
<td>0.556</td>
<td>0.556</td>
<td>0.556</td>
<td>0.556</td>
<td>0.556</td>
<td>0.556</td>
<td>0.556</td>
<td>0.556</td>
</tr>
<tr>
<td>32</td>
<td>0.805</td>
<td>0.804</td>
<td>0.805</td>
<td>0.804</td>
<td>0.805</td>
<td>0.804</td>
<td>0.805</td>
<td>0.804</td>
<td>0.805</td>
<td>0.805</td>
<td>0.804</td>
<td>0.805</td>
<td>0.805</td>
</tr>
<tr>
<td>64</td>
<td>0.902</td>
<td>0.901</td>
<td>0.902</td>
<td>0.901</td>
<td>0.902</td>
<td>0.901</td>
<td>0.902</td>
<td>0.901</td>
<td>0.902</td>
<td>0.902</td>
<td>0.901</td>
<td>0.902</td>
<td>0.902</td>
</tr>
</tbody>
</table>

Table B.3: Tolerance Interval Study Results for $\alpha = 0.95$.

<table>
<thead>
<tr>
<th>N</th>
<th>N(0,1)</th>
<th>N(0,10)</th>
<th>N(1,0)</th>
<th>N(1,10)</th>
<th>N(0,5)</th>
<th>N(0,10)</th>
<th>G(2,3)</th>
<th>G(5,6)</th>
<th>E(1)</th>
<th>B(6,4,0.7)</th>
<th>B(8,1)</th>
<th>CN(0,8,3)</th>
<th>CN(0,9,3)</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>0.946</td>
<td>0.9458</td>
<td>0.9382</td>
<td>0.9278</td>
<td>0.9752</td>
<td>0.9618</td>
<td>0.7684</td>
<td>0.847</td>
<td>0.6816</td>
<td>0.9154</td>
<td>0.999</td>
<td>0.9112</td>
<td>0.9382</td>
</tr>
<tr>
<td>16</td>
<td>0.9584</td>
<td>0.951</td>
<td>0.9348</td>
<td>0.9294</td>
<td>0.9854</td>
<td>0.9656</td>
<td>0.7052</td>
<td>0.7994</td>
<td>0.6308</td>
<td>0.9123</td>
<td>0.9966</td>
<td>0.907</td>
<td>0.9358</td>
</tr>
<tr>
<td>32</td>
<td>0.948</td>
<td>0.9348</td>
<td>0.9278</td>
<td>0.9264</td>
<td>0.9894</td>
<td>0.9658</td>
<td>0.6286</td>
<td>0.7428</td>
<td>0.5488</td>
<td>0.8748</td>
<td>1</td>
<td>0.9156</td>
<td>0.9414</td>
</tr>
<tr>
<td>64</td>
<td>0.956</td>
<td>0.9528</td>
<td>0.942</td>
<td>0.9346</td>
<td>0.9912</td>
<td>0.9626</td>
<td>0.5652</td>
<td>0.6954</td>
<td>0.4848</td>
<td>0.863</td>
<td>1</td>
<td>0.9302</td>
<td>0.957</td>
</tr>
<tr>
<td>96</td>
<td>0.9512</td>
<td>0.9464</td>
<td>0.943</td>
<td>0.9386</td>
<td>0.992</td>
<td>0.9748</td>
<td>0.5048</td>
<td>0.6266</td>
<td>0.4314</td>
<td>0.8036</td>
<td>1</td>
<td>0.954</td>
<td>0.965</td>
</tr>
</tbody>
</table>

Table B.2: Tolerance Interval Study Results for $\alpha = 0.8$. 
B.4 Proof for Maximum Tolerance Interval

The proof of the maximum tolerance interval can be best viewed as a discrete implementation of the probability function. Let $U_{ij}$ represent the $j$th calculation of the upper bound using the downsampling tolerance interval method. Let $U_{2j}$ represent the $j$th calculation of the upper bound using the tolerance interval method based on the normality assumption. Then

$$P(F(U_1) > p_1) = \frac{\sum_{j=1}^{N} T(F(U_{ij}) > p_1)}{N} \text{ as } N \to \infty,$$

and

$$P(F(U_2) > p_1) = \frac{\sum_{j=1}^{N} T(F(U_{2j}) > p_1)}{N} \text{ as } N \to \infty,$$

where the function $T(x)$ returns a 1 if $x$ is true and a 0 if $x$ is false. Then since individually,

$$T[F(\max(U_{1j}, U_{2j})) > p_1] \geq T[F(U_{1j}) > p_1],$$
$$T[F(\max(U_{1j}, U_{2j})) > p_1] \geq T[F(U_{2j}) > p_1].$$

then the summation across $j$

$$P(F(\max(U_1, U_2)) > p_1) \geq P(F(U_1) > p_1),$$
$$P(F(\max(U_1, U_2)) > p_1) \geq P(F(U_2) > p_1)$$

holds. It can be seen that

$$P(F(\max(U_1, U_2)) > p_1) \geq \max(P(F(U_1) > p_1)), P(F(U_2) > p_1).$$

B.5 Code for Normal Assumption Tolerance Interval Calculation

This code can be used to calculate a tolerance interval using the normality assumption.

```matlab
function [interval, interval_pi] = calc_tolerance_interval_normal(beta_val, gamma_val, x, one_sided)

% This function calculates the tolerance interval based on the data
% the population percentage (beta),
% and the confidence level (gamma), and whether it is one-sided.
% Formula from NIST website. This is valid for a one side or two-sided test.
% This function also calculates mean prediction intervals.
```
% Assumes normally distributed data.
N = numel(x);
sample_mean = mean(x);
sample_std = std(x);
if nargin < 4
    one_sided = 0;
end
nu = N - 1;
if (one_sided == 0)
    % Two sided test
    chi_sqr = chi2inv(1-gamma_val,nu);
z_sqr = norminv((1-beta_val)/2).^2;
k = sqrt(nu*(1+1/N)*z_sqr/chi_sqr);
    delta = k*sample_std;
    interval = [sample_mean - delta sample_mean + delta];
    delta_pi = -tinv((1-gamma_val)/2,nu)*sample_std*sqrt(1+1/N);
    interval_pi = [sample_mean - delta_pi sample_mean + delta_pi];
else
    % One sided test.
    delta = norminv(beta_val)*sqrt(N);
    % This line is the most computationally intensive.
    k = nctinv(gamma_val,N-1,delta)/sqrt(N);
    delta = k*sample_std;
    interval = sample_mean + delta;
    delta_pi = -tinv((1-gamma_val),nu)*sample_std*sqrt(1+1/N);
    interval_pi = [sample_mean + delta_pi];
end

B.6 Code for Non-Parameter Tolerance Interval Calculation Baseline.

This code can be used to calculate the non-parametric tolerance interval for the baseline method, and the max points method.

function [tol_interval, reported_gamma, r, s] = ...
calc_nonparam_tolerance_interval(beta, gamma, x, one_sided)
% This function calculates a non-parametric tolerance interval for the data in x based on the Wilks method. "beta" references the population percentage, "gamma" references the probability of the population having a proportion of data within the "tol_interval". Setting the input "one_sided" to a 1 indicates desire to calculate a one sided interval. If one sided, tol_interval(1) has the value for the low side only interval, and tol_interval(2) has the value for a high side only interval.

% References
% - Wilks, S.S., Determination of Sample Sizes for setting tolerance limits, Annals of Mathematical Statistics, 12, 91-96 (1941)

%beta = 0.99; % Population percentage
%gamma = 0.95; % Confidence that the population is within p
a_sorted = sort(x);
um_pts = length(x);
% Given a number of data points which indices should be selected.
% r is the first index used for calculation of the interval.
% s is the upper limit used for calculation of the interval.
if one_sided
    num_pts = 1;
    r = 1;
    s = num_pts+1;
    test_gamma = 1 - betacdf(beta,s-r,num_pts-s+r+1);
    for k = floor(num_pts/2):-1:1
        r = k;
        s = num_pts+1;
        test_gamma = 1 - betacdf(beta,s-r,num_pts-s+r+1);
        if test_gamma > gamma
            break;
        end
    end
tol_interval(1) = a_sorted(r);
reported_gamma(1) = test_gamma;
for k = floor(num_pts/2):-1:1
    r = 0;
    s = num_pts-k+1;
    test_gamma = 1 - betacdf(beta,s-r,num_pts-s+r+1);
    if test_gamma > gamma
        break;
    end
end
tol_interval(2) = x_sorted(s);
reported_gamma(1) = test_gamma;
else
    for k = floor(num_pts/2):-1:1
        r = k;
        s = num_pts-k+1;
        test_gamma = 1 - betacdf(beta,s-r,num_pts-s+r+1);
        if test_gamma > gamma
            break;
        end
    end
disp([r s])
tol_interval = [x_sorted(r) x_sorted(s)];
reported_gamma = test_gamma;
end

B.7 Code for Non-Parameter Tolerance Interval Calculation with Downsampling.

function [tol_interval, reported_gamma, r, s] = ...
calc_nonparam_tolerance_interval_v2(beta, gamma, x, one_sided,num_avgs)
% This function calculates a non-parametric tolerance interval for the % data in x based on the Wilks method. "beta" references the population % percentage, "gamma" references the probability of the population having % a proportion of data within the "tol_interval". % Setting the input "one_sided" to a 1 indicates desire to calculate a one % sided interval. If one sided, tol_interval(1) has the value for the low % side only interval, and tol_interval(2) has the value for a high side % only interval.
% This function was an attempt to average data by selecting the numbering % of averages to use. In the end did not do a good job of reducing beta % for data sets that were larger than what was needed. Recommend setting % it to 1.
% This function does downsample a subset of the data of x to obtain a % tolerance interval with probability gamma. Does not yet work for two % sided samples.
% References
% - Wilks, S.S., Determination of Sample Sizes for setting tolerance % limits, Annals of Mathematical Statistics, 12, 91-96 (1941)
%beta = 0.99; % Population percentage
%gamma = 0.95; % Confidence that the population is within p
x_sorted = sort(x);
num_pts = length(x);
% Given a number of data points which indices should be selected.
% r is the first index used for calculation of the interval
% s is the the upper limit used for calculation of the interval.
if one_sided

% Lower side calculation
for k = floor(num_pts/2):-1:1
    r = k;
    s = num_pts-s;
    test_gamma = 1 - betacdf(beta,s-r,num_pts-s+r+1);
    if test_gamma > gamma
        break;
    end
end
tol_interval(1) = x_sorted(r);
reported Gamma(1) = test_gamma;
% Upper side calculation
for k = floor(num_pts/2):-1:1
    r = 0;
    s = num_pts-k+1;
    test Gamma = 1 - betacdf(beta,s-r,num_pts-s+r+1);
    if test Gamma > gamma
        % Need to determine minimum number of points needed for a % tolerance interval reduce num_pts until test Gamma < gamma
        for indx_min_len = num_pts:-1:k
            s_temp = indx_min_len - k + 1;
        end
    end
test_gamma2 = 1 - betacdf(beta,s_temp-r,indx_min_len - s_temp + r + 1);
if test_gamma2 < gamma
   break;
end

% indx_min_len+1 is the minimum length
if indx_min_len+1 < num_pts
   num_avgs = 1;
   tol_interval_temp = zeros(2,num_avgs);
   for indx_avg = 1:num_avgs
      x_rand = datasample(x,indx_min_len+1,'Replace',false);
      tol_interval_temp(:,indx_avg) = ...
         calc_nonparam_tolerance_interval(beta, gamma, x_rand, one_sided);
   end
tol_interval(2) = mean(tol_interval_temp(2,:));
   s = indx_min_len+1;
else
   tol_interval(2) = x_sorted(s);
end

% Two sided test
% The two sided test needs more work to implement the re-sampling.
for k = floor(num_pts/2):-1:1
   r = k;
   s = num_pts-k+1;
   test_gamma = 1 - betacdf(beta,s-r,num_pts-s+r+1);
   if test_gamma > gamma
      break;
   end
end
disp([r s])
tol_interval = [x_sorted(r) x_sorted(s)];
reported_gamma = test_gamma;

B.8 Code for Simulation Study

clear all;
close all;

% Use the default random number generator and seed with a 1 for debug purposes.
%rng('default')
%rng('shuffle');

% This routine is based on the paper Better Confidence Intervals for the Variance in a Random Sample by Hummel. Instead of confidence intervals for variance, it is calculating tolerance intervals. The same distributions are used for testing.

% Test nonparametric tolerance interval
plot_fig = 0;
pd{1} = makedist('Normal','mu',0,'sigma',10);
pd{2} = makedist('Normal','mu',80,'sigma',10);
pd{3} = makedist('tLocationScale','mu',0,'sigma',1,'nu',10);
pd{4} = makedist('tLocationScale','mu',0,'sigma',1,'nu',5);
pd{5} = makedist('Uniform','lower',0,'upper',1);
pd{6} = makedist('Beta','a',3,'b',3); % variance = 0.0357, std = 0.1890
pd{7} = makedist('Beta','a',8,'b',1); % variance = 0.0357, std = 0.1890
pd{8} = makedist('Gamma','a',2,'b',3);
pd{9} = makedist('Exponential','mu',1);
pd{10} = makedist('Beta','a',0.4,'b',0.7); % variance = 0.0357, std = 0.1890
pd{11} = makedist('Beta','a',0.8,'b',1.1); % variance = 0.0357, std = 0.1890

distr_cn = obtain_contaminated_normal_data(0.8,0.5,[500000,1]);
[distr_cn_F,distr_cn_x] = ksdensity(distr_cn,'npoints',200,'function','cdf');
pd{12}.DistributionName = 'CNormal';
pd{12}.lambda = 0.8;
pd{12}.sigma = 3;
pd{12}.dist_cn_F = distr_cn_F;
pd{12}.dist_cn_x = distr_cn_x;

distr_cn = obtain_contaminated_normal_data(0.9,0.5,[500000,1]);
[distr_cn_F,distr_cn_x] = ksdensity(distr_cn,'npoints',200,'function','cdf');
pd{13}.DistributionName = 'CNormal';
pd{13}.lambda = 0.9;
\[
p_{d\{13\}}.sigma = 3;
\]
\[
p_{d\{13\}}.dist\_cn\_F = dist\_cn\_F;
\]
\[
p_{d\{13\}}.dist\_cn\_x = dist\_cn\_x;
\]
\[
num\_c\_too\_high = 0;
\]
\[
num\_c2\_too\_high = 0;
\]
\[
num\_c3\_too\_high = 0;
\]
\[
% The statistic of interest is such that
% \( P( (F(U) - F(L)) \geq beta) \geq gamma \)
\]
\[
beta\_val = 0.95; \quad \% \text{population percentage}
\]
\[
gamma\_val = 0.8; \quad \% \text{probability}
\]
\[
num\_reps = 5000;
\]
\[
num\_reps = 10;
\]
\[
um\_distr = 13;
\]
\[
um\_samples = 50;
\]
\[
num\_sides = 1;
\]
\[
num\_samples\_test = [8 16 32 50 76];
\]
\[
um\_samples\_test = [76];
\]
\[
len\_sample\_test = numel(num\_samples\_test);
\]
\[
gamma\_np = zeros(len\_sample\_test, num\_distr);
\]
\[
gamma\_np_1 = zeros(len\_sample\_test, num\_distr);
\]
\[
gamma\_np_1000 = zeros(len\_sample\_test, num\_distr);
\]
\[
gamma\_norm = zeros(len\_sample\_test, num\_distr);
\]
\[
gamma\_max = zeros(len\_sample\_test, num\_distr);
\]
\[
converged = zeros(len\_sample\_test, num\_distr, num\_reps);
\]
\[
coverage\_np_1000\_logged = zeros(len\_sample\_test, num\_distr, num\_reps);
\]
\[
tolerance\_interval\_np\_1\_logged = zeros(len\_sample\_test, num\_distr, num\_reps);
\]
\[
tolerance\_interval\_np\_1000\_logged = zeros(len\_sample\_test, num\_distr, num\_reps);
\]
\[
tolerance\_interval\_norm\_logged = zeros(len\_sample\_test, num\_distr, num\_reps);
\]
\[
tolerance\_interval\_max\_np\_norm = zeros(len\_sample\_test, num\_distr, num\_reps);
\]
\[
tolerance\_interval\_gmm\_logged = zeros(len\_sample\_test, num\_distr, num\_reps);
\]
\[
\]
\[
% This for statement loops through different sample sizes
\]
\[
for indx\_num\_samples = 1:len\_sample\_test
\]
\[
num\_samples = num\_samples\_test(indx\_num\_samples)
\]
\[
% This for statement loops through the different distribution types.
\]
\[
for indx\_distr = 1:num\_distr
\]
\[
for indx\_distr = 1:1:
\]
\[
np\_indx = 0;
\]
\[
norm\_indx = 0;
\]
\[
np\_1\_indx = 0;
\]
\[
np\_1000\_indx = 0;
\]
\[
max\_indx = 0;
\]
\[
gmm\_indx = 0;
\]
\[
% Simulate \text{num\_reps} times the calculation of the tolerance interval.
\]
\[
for indx = 1:num\_reps
\]
\[
if strcmp(pd(indx\_distr).DistributionName,'CNormal');
\]
\[
data\_sample = obtain\_contaminated\_normal\_data ...
\]
\[
(pd(indx\_distr).lambda,pd(indx\_distr).sigma,[num\_samples,1]);
\]
\[
else
\]
\[
data\_sample = random(pd(indx\_distr),[num\_samples,1]);
\]
\[
end
\]
\[
if (1)
\]
\[
% Calculate a non-parametric tolerance interval. baseline
\]
\[
[tol\_interval\_np, act\_gamma, r, s] = ...
\]
\[
calc\_nonparam\_tolerance\_interval(beta\_val, gamma\_val, data\_sample, (num\_sides == 1));
\]
\[
% Calculate a non-parametric tolerance interval.
\]
\[
% Downsampling method.
\]
\[
[tol\_interval\_np\_1, act\_gamma\_1] = ...
\]
\[
calc\_nonparam\_tolerance\_interval\_v2(beta\_val, gamma\_val, data\_sample, (num\_sides == 1), 1);
\]
\[
% Calculate a non-parametric tolerance interval. All points
\]
\[
% method.
\]
\[
[tol\_interval\_np\_1000, act\_gamma\_1000] = ...
\]
\[
calc\_nonparam\_tolerance\_interval(beta\_val, 1, data\_sample, (num\_sides == 1));
\]
% Calculate a tolerance interval based on assumption the distribution is normal.
[tol_interval_norm,delta] = ...
calc_tolerance_interval(mean(data_sample),std(data_sample),length(data_sample),beta_val,gamma_val,num_sides);

% Calculate tolerance interval with a gaussian mixture model
[tolerance_interval_gmm,converged(indx_num_samples,indx_distr,indx)] = ...
calc_tol_interval_w_gmm(data_sample,beta_val,gamma_val);
else
tol_interval_np_1(2) = 0;
% Calculate a non-parametric tolerance interval. (with all pts).
[tol_interval_np_1000, act_gamma_1000] = ...
calc_nonparam_tolerance_interval(beta_val, 1, data_sample, (num_sides == 1));
act_gamma = act_gamma_1000;
tol_interval_np(2) = 0;
tolerance_interval_gmm = 0;
end
tolerance_interval_np_1_logged(indx_num_samples,indx_distr,indx) = tol_interval_np_1(2);
tolerance_interval_np_1000_logged(indx_num_samples,indx_distr,indx) = tol_interval_np_1000(2);
tolerance_interval_np_logged(indx_num_samples,indx_distr,indx) = tol_interval_np(2);
tolerance_interval_norm_logged(indx_num_samples,indx_distr,indx) = tol_interval_norm;
tolerance_interval_gmm_logged(indx_num_samples,indx_distr,indx) = tolerance_interval_gmm;

% Added additional logic to use the parametric free if it has sufficient actual gamma.
% disp(act_gamma)
if (act_gamma < gamma_val)
tolerance_interval_max_np_norm(indx_num_samples,indx_distr,indx) = max(tol_interval_np_1(2),tol_interval_norm);
else
tolerance_interval_max_np_norm(indx_num_samples,indx_distr,indx) = tol_interval_np_1(2);
end
if (plot_fig)
figure;
hist(data_sample);
axisValues = axis;
hold on;
plot([tol_interval_np(1) tol_interval_np(1)],[axisValues(3) axisValues(4)],'r--');
plot([tol_interval_np(2) tol_interval_np(2)],[axisValues(3) axisValues(4)],'r--');
hold off;
end
% Do something different if the tolerance interval is one sided versus two sided. For one-sided intervals, take directly the upper limit \( \Pr(U) \geq \beta \geq \gamma \) vs. a two sided limit \( \Pr((U-L) \geq \beta) \geq \gamma \).
if num_sides == 1
% Matlab does not natively support the contaminated normal distribution so the distribution is created from 500,000 samples.
if strcmp(pd{indx_distr}.DistributionName,'CNormal');
coverage_np = interp1(pd{indx_distr}.dist_cn_x,pd{indx_distr}.dist_cn_F, ...
    tol_interval_np(2));
coverage_norm = interp1(pd{indx_distr}.dist_cn_x,pd{indx_distr}.dist_cn_F, ...
    tol_interval_norm);
coverage_np_1 = interp1(pd{indx_distr}.dist_cn_x,pd{indx_distr}.dist_cn_F, ...
    tol_interval_np_1(2));
coverage_np_1000 = interp1(pd{indx_distr}.dist_cn_x,pd{indx_distr}.dist_cn_F, ...
    tol_interval_np_1000(2));
coverage_max = interp1(pd{indx_distr}.dist_cn_x,pd{indx_distr}.dist_cn_F, ...
    tolerance_interval_max_np_norm(indx_num_samples,indx_distr,indx));
coverage_gmm = interp1(pd{indx_distr}.dist_cn_x,pd{indx_distr}.dist_cn_F, ...
    tolerance_interval_max_np_norm(indx_num_samples,indx_distr,indx));
else
coverage_np = cdf(pd{indx_distr},tol_interval_np(2));
coverage_norm = cdf(pd{indx_distr},tol_interval_norm);
coverage_np_1 = cdf(pd{indx_distr},tol_interval_np_1(2));
coverage_np_1000 = cdf(pd{indx_distr},tol_interval_np_1000(2));
coverage_max = cdf(pd{indx_distr}, ...
    tolerance_interval_max_np_norm(indx_num_samples,indx_distr,indx));
coverage_gmm = cdf(pd{indx_distr},tolerance_interval_gmm);
end
else
if strcmp(pd{indx_distr}.DistributionName,'CNormal');
coverage_np = interp1(pd{indx_distr}.dist_cn_x,pd{indx_distr}.dist_cn_F, ...
    tol_interval_np(1) - interp1(pd{indx_distr}.dist_cn_x,pd{indx_distr}.dist_cn_F,tol_interval_np(1));
coverage_norm = interp1(pd{indx_distr}.dist_cn_x,pd{indx_distr}.dist_cn_F,tol_interval_norm(2) - interp1(pd{indx_distr}.dist_cn_x,pd{indx_distr}.dist_cn_F,tol_interval_norm(1));
else
coverage_np = cdf(pd{indx_distr},tol_interval_np(2)) - cdf(pd{indx_distr},tol_interval_np(1));
coverage_norm = cdf(pd{indx_distr},tol_interval_norm(2)) - cdf(pd{indx_distr},tol_interval_norm(1));
end
end
end
coverage_np_1000_logged(indx_num_samples, indx_distr, indx) = coverage_np_1000;
coverage_np_1_logged(indx_num_samples, indx_distr, indx) = coverage_np_1;
coverage_norm_logged(indx_num_samples, indx_distr, indx) = coverage_norm;
coverage_max_logged(indx_num_samples, indx_distr, indx) = coverage_max;
coverage_gmm_logged(indx_num_samples, indx_distr, indx) = coverage_gmm;

% Calculate the number of times the tolerance interval included the
% beta amount of the population.
if (coverage_np > beta_val)
    np_indx = np_indx + 1;
end
if (coverage_norm > beta_val)
    norm_indx = norm_indx + 1;
end
if (coverage_np_1 > beta_val)
    np_1_indx = np_1_indx + 1;
end
if (coverage_np_1000 > beta_val)
    np_1000_indx = np_1000_indx + 1;
end
if (coverage_max > beta_val)
    max_indx = max_indx + 1;
end
if (coverage_gmm > beta_val)
    gmm_indx = gmm_indx + 1;
end

gamma_np(indx_num_samples, indx_distr) = np_indx / num_reps;
gamma_norm(indx_num_samples, indx_distr) = norm_indx / num_reps;
gamma_np_1(indx_num_samples, indx_distr) = np_1_indx / num_reps;
gamma_np_1000(indx_num_samples, indx_distr) = np_1000_indx / num_reps;
gamma_max(indx_num_samples, indx_distr) = max_indx / num_reps;
gamma_gmm(indx_num_samples, indx_distr) = gmm_indx / num_reps;

end

% xlswrite('temp.xls', [gamma_np; gamma_np_1; gamma_np_1000; gamma_norm; gamma_max; gamma_gmm])
APPENDIX C

PROOFS FOR TECHNICAL APPROACH AND GENERALIZED SOLUTION

C.1 Negative Triangle Inequality Proof

First demonstrate that

$$||x|| - ||y|| \leq ||x - y||$$

$$||x - y + y|| = ||x - y + y||$$

$$||x|| = ||x - y + y||$$

$$||x|| \leq ||x - y|| + ||y||$$ by the triangle inequality

$$||x|| - ||y|| \leq ||x - y||$$

C.2 Quadratic Inequality

Note, the last two steps in the Lyapunov stability proof in Section 5.3.1 are simplified through

$$a^2 + 2ab + b^2 \geq 0$$

$$b^2 \geq -a^2 + 2ab$$
For example,

\[ a = |e_1| \]
\[ b = \frac{1}{2} \Delta_1 \]
\[ \frac{1}{4} \Delta_1^2 \geq -e_1^2 + \Delta_1 |e_1| \]

and

\[ a = \sqrt{k_{20} \beta_e |e_2|} \]
\[ b = \frac{1}{2 \sqrt{k_{20}}} \beta_e (\Delta_2 + W_3 \Delta_1) \]
\[ \frac{1}{4k_{20}} \beta_e^2 (\Delta_2 + W_3 \Delta_1)^2 \geq -\beta_e^2 k_{20} e_2^2 + \beta_e^2 (\Delta_2 + W_3 \Delta_1) |e_2| \]

**C.3 Nonlinear Damping**

The nonlinear damping backstepping control algorithm development is modified slightly from that found in [44]. The control algorithm here includes the reference signals and is shown below.

\[ \alpha = 1/g_1(-f_1 - k_1 e_1 - c_1 \Delta_1^2 e_1), \]  
(C.1)

\[ u = 1/g_2(-f_2 + \hat{\alpha} - k_2 e_2 - g_1 e_1 - c_2 \Delta_2^2 e_2 - \omega e_2), \]  
(C.2)

where

\[ \omega = c_3 (\frac{\partial \alpha}{\partial \Delta_1})^2 \]  
(C.3)

and the constant \( \Delta_1 \geq |\delta_1|, \Delta_2 \geq |\delta_2|, c_1, c_2, \) and \( c_3 \) are \( \mathbb{R}^+ \) and known. To show boundedness, the Lyapunov candidate for \( e_1 \) used is \( V_1 = 1/2 e_1^2 \). The following Lyapunov analysis shows the
stability quality of the first stage

\[ \dot{V}_1 = e_1 \dot{e}_1 = e_1(-k_1 e_1 + g_1 e_2 + \delta_1 - c_1 \Delta_1 e_1) \]

\[ = -k_1 e_1^2 + g_1 e_1 e_2 + \delta_1 e_1 - c_1 \Delta_1 e_1^2 \]

\[ \leq -k_1 e_1^2 + g_1 e_1 e_2 + \Delta_1 |e_1| - c_1 \Delta_1 e_1^2 \]

\[ \leq -k_1 e_1^2 + g_1 e_1 e_2 + \frac{1}{4c_1}. \]

The Lyapunov candidate for the system is

\[ V = V_1 + \frac{1}{2} e_2^2 \]

and

\[ \dot{V} \leq -k_1 e_1^2 + g_1 e_1 e_2 + \frac{1}{4c_1} + e_2(\dot{x}_2 - \dot{\alpha}) \]

\[ \leq -k_1 e_1^2 + \frac{1}{4c_1} + g_1 e_1 e_2 \]

\[ + e_2(-k_2 e_2 + \dot{\alpha} - g_1 e_1 - c_2 \Delta_2 e_2 - \omega e_2 - \dot{\alpha} + \delta_2) \]

\[ \leq -k_1 e_1^2 - k_2 e_2^2 + \frac{1}{4c_1} - e_2 \frac{\partial \alpha}{\partial x_1} \delta_1 - \omega e_2^2 \]

\[ + \delta_2 e_2 - c_2 \Delta_2 e_2^2 \]

\[ \leq -k_1 e_1^2 - k_2 e_2^2 + \frac{1}{4c_1} + \frac{\partial \alpha}{\partial x_1} |e_2| - \omega e_2^2 \]

\[ + \Delta_2 |e_2| - c_2 \Delta_2 e_2^2 \]

\[ \leq -k_1 e_1^2 - k_2 e_2^2 + \frac{1}{4c_1} + \frac{1}{4c_2} + \frac{1}{4c_3}. \]

One can then show that \( e_1 \) is bounded by [44]

\[ \lim_{t \to \infty} |e_1(t)| \leq \sqrt{\frac{2W_d}{k_m}}. \]

where

\[ W_d = \frac{1}{4c_1} + \frac{1}{4c_2} + \frac{1}{4c_3}, \]

\[ k_m = \min(k_1, k_2). \]
The nonlinear damping control law was not implemented as the benefits of this type of control law over classical feedback linearization were not experienced due to the limits on the error gains due to the delay found with the Speedgoat experimental system. A few simulation studies were performed without the delays, and show that higher gains could be realized and the effective uniform ultimate bound could be reduced with nonlinear damping control.
APPENDIX D

DIODE DROP EXPLANATION

Based on the work in [64] a model of the second order diode characteristic was created. The zeroth order and first order characteristics (i.e., diode resistive and diode forward voltage drop) do not change. The second order characteristics follow a similar build-up. First determination of an average current value is found

\[ I_{DF} = \left( \frac{1}{T} \int_0^T i_d(t)^3 dt \right)^{\frac{1}{3}}, \]

where \( T \) is the inverse of the switching frequency, \( i_d \) is the current through the diode which is

\[ i_d(t) = \begin{cases} 0 & : 0 < t \leq DT \\ I_L & : DT < t \leq T \end{cases} \]

where \( D \) is the duty cycle. Solving the integral results in

\[ I_{DF} = I_L (1 - D)^{\frac{1}{3}}. \]

The power loss across the diode for the second order characteristic is

\[ P_{DF} = D_F I_{DF}^3 = D_F I_L^3 (1 - D) \]

and the voltage drop is

\[ V_{DF} = D_F I_L^2 (1 - D). \]
E.1 Method of Linear Least Squares

The method of linear least squares is used throughout this dissertation and is used to identify the model and control uncertainty. Typically, when used in this dissertation it is used with state variable filtering. To combine the state variable filtering (see Eq. (2.3)) with the uncertainty calculation (see Eq. (4.3) and (4.4))

\[
\frac{\lambda\delta_{c1}}{p + \lambda} = \frac{\lambda px_1}{p + \lambda} - \frac{\lambda px_{c1}}{p + \lambda}, 
\]

\[
\frac{\lambda\delta_{c2}}{p + \lambda} = \frac{\lambda px_2}{p + \lambda} - \frac{\lambda px_{c2}}{p + \lambda}. 
\]

(E.1)  \hspace{1cm}  (E.2)

For the dissertation work, a polynomial model will be used as the uncertainty estimator. Other techniques such as neural networks or fuzzy approximators could be used [44], [82], [85]. Though, calculation of the Lipschitz constant and distance function may be more difficult. For example, if the polynomial basis function is assumed to be second order then the following holds:

\[
\delta_{c1} = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1 x_2 + \theta_4 x_1^2 + \theta_5 x_2^2 + O_1(X^3),
\]

\[
\frac{\lambda\delta_{c1}}{p + \lambda} = \theta_0 + \theta_1 \frac{\lambda x_1}{p + \lambda} + \theta_2 \frac{\lambda x_2}{p + \lambda} + \theta_3 \frac{\lambda x_1 x_2}{p + \lambda} + \theta_4 \frac{\lambda x_1^2}{p + \lambda} + \theta_5 \frac{\lambda x_2^2}{p + \lambda} + \frac{\lambda O_1(X^3)}{p + \lambda}, 
\]

(E.3)
Setting equation (E.1) to (E.3), the parameter $\theta$ can be found through linear least squares that leads to the following uncertainty models

$$\hat{\delta}_{c1} = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1 x_2 + \theta_4 x_1^2 + \theta_5 x_2^2,$$

For example, using the main factor terms model only (i.e., using $V_o, V_i, I_o, \dot{I}_o$) and using state variable filtering to identify the control uncertainty $\delta_{c1}$,

$$\frac{\lambda \delta_{c1}}{p + \lambda} = \theta_0 + \theta_1 \frac{\lambda V_o}{p + \lambda} + \theta_2 \frac{\lambda V_i}{p + \lambda} + \theta_3 \frac{\lambda I_o}{p + \lambda} + \theta_4 \frac{\lambda p I_o}{p + \lambda},$$

where

$$V_{po} = \frac{\lambda V_o}{p + \lambda},$$

$$V_{pi} = \frac{\lambda V_i}{p + \lambda},$$

$$I_{po} = \frac{\lambda I_o}{p + \lambda},$$

$$I_{pdoto} = \frac{\lambda p I_o}{p + \lambda}.$$

The data is built up such that for each experimental run a time slice is taken from that run ($0.05 < t < 0.11$). So that $V_{po}(n, m)$ refers to the $n^{th}$ test of $N$ tests within a data set and $m^{th}$ data point of $M$ data points within that test, then the regressor, $X$, is

$$X = \begin{bmatrix}
1 & V_{po}(1, 1) & V_{pi}(1, 1) & I_{po}(1, 1) & I_{pdoto}(1, 1) \\
1 & V_{po}(1, 2) & V_{pi}(1, 2) & I_{po}(1, 2) & I_{pdoto}(1, 2) \\
\vdots & \vdots & \vdots & \vdots & \vdots \\
1 & V_{po}(1, M) & V_{pi}(1, M) & I_{po}(1, M) & I_{pdoto}(1, M) \\
1 & V_{po}(2, 1) & V_{pi}(2, 1) & I_{po}(2, 1) & I_{pdoto}(2, 1) \\
\vdots & \vdots & \vdots & \vdots & \vdots \\
1 & V_{po}(N, M) & V_{pi}(N, M) & I_{po}(N, M) & I_{pdoto}(N, M)
\end{bmatrix}.$$
E.2 Parsimonious Models

Using linear least squares by itself with quadratic polynomial models will identify the parameters for each of the basis functions. Unfortunately, those parameters and basis functions may not be an optimal selection (i.e., one can find a model with fewer terms that has the same performance). It was found that the Lipschitz calculations relies on identifying parsimonious models.

There are a number of means of identifying an optimal polynomial model [50], the first is forward selection, the second is backward elimination, and the third is a combination of the two. For all three methods, the F-statistic is used to evaluate whether a term is significant. In the forward selection method, each term is evaluated and if it is deemed significant via the F-statistic, it is added to the model. In backward elimination, the model starts with all terms and the “worst term” (i.e., the one with the least significance) is removed until only those terms that are significant are left. Stepwise starts with the forward selection idea and allows terms in with a certain significance, but after a term is added, an additional check is made to remove those terms that may not be significant. For a longer discussion on the different means to perform this model optimization see [50].

Work was done with Mathwork’s implementation of stepwise, but using time series data causes most terms to be deemed significant because the degrees of freedom is equal to the total number of points used. So instead, a separate function was created that made use of backward elimination only, but used for degrees of freedom in the F-statistic analysis the number of different boundary conditions tested rather than the number of points used in the regression. Effectively, the number of tests $N$ above was used as the degrees of freedom. This change aided in identifying the parsimonious models and was tested against the example problems and demonstrated superior performance over Mathwork’s implementation. Mathwork’s code often selected the model with all terms. The code used for this work can be found in Section E.3.
E.3 Code for Stepwise Function

```matlab
function mdl = stepwise_jtz(ds,model_type,penter,premove,alt_dF)
% This implementation of stepwise is primarily based on backward
% elimination only. The work here that goes beyond what typically is
% found, is the allowance to change the degrees of freedom. For time
% series based regression analysis this is key, in finding parsimonious
% models
% ds - assumes last data is the output. Matlab data set.
% model_type - linear, interactions, or quadratic.
% Assumes the ds inputs are the linear versions, i.e., if a quadratic
% model_type is used, it will square those terms.
% penter - not used.
% premove - p value to remove terms.
% alt_dF - alternative degrees of freedom.
%
% For Debug
%ds = mat2dataset([x1' x2' x3' x4' y'],'VarNames',{'x1','x2','x3','x4','Y'});
ds = mat2dataset([x1' x2' x3' y'],'VarNames',{'x1','x2','x3','Y'});
ds = mat2dataset([x1' x2' y'],'VarNames',{'x1','x2','Y'});
% Be Cautious there is slightly different behaviour between debug mode vs.
% not in debug mode when setting dF.
% ds = ds_w1;
%alt_dF = 64;
%model_type = 'interactions';
penter = 0.05;
pmove = 0.10;
% % dF = num_runs;
% n = length(ds);
%alt_dF = 11
%
data = double(ds);
[n,num_coefs_regr_temp] = size(data);
if nargin < 2
penter = 0.05;
pmove= 0.1;
elseif nargin > 4
dF = alt_dF;
else
 dF = n;
end
dF = alt_dF;
x_var_pre_scaled = data(:,1:end-1);
y_var = data(:,end);
%
% Construct full model based on model selection
if strcmp(model_type,'linear')
 Data_TF = x2fx(x_var,'linear');
 var_table = [zeros(1,num_regr)*eye(num_regr,num_regr)];
else if strcmp(model_type,'interactions')
 Data_TF = x2fx(x_var,'interaction');
 var_table = [zeros(1,num_regr)*eye(num_regr,num_regr)];
for indx = 1:num_regr-1
 var_table = [var_table;zeros(num_regr-indx,indx-1) ones(num_regr-indx,1) eye(num_regr-indx,num_regr-indx)];
end
end
```

182
elseif strcmp(model_type,'quadratic')
    Data_TF = x2fx(x_var,'quadratic');
    %Data_TF = calc_regressor_derivatives(Data_TF_unfilt,filter_a0,sample_time);
    var_table = [zeros(1,num_regr);eye(num_regr,num_regr)];
    for indx = 1:num_regr-1
        var_table = [var_table;zeros(num_regr-indx,indx-1) ones(num_regr-indx,1) eye(num_regr-indx,num_regr-indx)];
    end
    var_table = [var_table;eye(num_regr,num_regr)+2];
end

% Run the first model with all terms
mdl_temp = calc_stats(y_var,Data_TF,dF);
new_model = var_table;
old_model = new_model;

% Remove non-main terms with largest p > premove one at a time (starting with the worse)
while ~isempty(find(mdl_temp.p_f_coef(num_regr+2:end)>premove,1))
    [~,indx] = sort(mdl_temp.p_f_coef(num_regr+2:end));
    % Remove the rows
    new_model = [old_model(1:indx(end)-1,:);old_model(indx(end)+1:end,:)];
    Data_TF = x2fx(x_var,new_model);
    %Data_TF = calc_regressor_derivatives(Data_TF_unfilt,filter_a0,sample_time);
    mdl_temp = calc_stats(y_var,Data_TF,dF);
    old_model = new_model;
    % disp(old_model)
    % disp(mdl_temp)
    % mdl_temp.Fo_coef
    % mdl_temp.SSr_coef
    % mdl_temp.p_f_coef
end

% Remove main terms that may no longer be needed.
% Check to see if a main term is used more than once.
    indx_child = [];
    indx_child_temp = find(sum(fliplr(old_model),1)==1);
    for temp_indx = 1:length(indx_child_temp)
        row_val = [zeros(1,num_regr-indx_child_temp(temp_indx)) 1 zeros(1,indx_child_temp(temp_indx)-1)];
        [~,indx_child(temp_indx)] = intersect(old_model, ... row_val,'rows');
    end
% There may be an issue with removing a main term that no longer is needed,
% that than becomes a non-main term that is no longer needed.
% Remove only those main terms that have p values > premove ... one at a
% time. Then re-check.
while ~isempty(find(mdl_temp.p_f_coef(indx_child)>premove,1))
    [~,indx_list] = sort(mdl_temp.p_f_coef(indx_child));
    indx_child = indx_child(indx_list); 
    new_model = [old_model(1:indx(end)-1,:);old_model(indx(end)+1:end,:)];
    Data_TF = x2fx(x_var,new_model);
    %Data_TF = calc_regressor_derivatives(Data_TF_unfilt,filter_a0,sample_time);
    % re-calculate new models parameter coefficients.
    mdl_temp = calc_stats(y_var,Data_TF,dF);
    old_model = new_model;
    % disp(mdl_temp)
    % disp(indx_child)
end

% Add in terms that were removed one at a time allowing to enter if p < penter
% any time a term has p > premove then remove the term.
% Could use the function addTerms
% Start copying over model parameters
mdl = mdl_temp;
mdl.beta_normalized = mdl_temp.beta;
mdl.model_desc = new_model;

% Need to convert from normalized factors back to engineering terms
offset_gain_vector = x_var_offset./x_var_gain;
[mdl.NumCoefficients, ''] = size(new_model);
gamma = zeros(mdl.NumCoefficients,1);

% This section converts the scaled terms to non-scaled terms.
% The first term is the intercept term.
gamma(1) = x2fx(offset_gain_vector,new_model)*mdl_temp.beta;
for indx = 2:mdl.NumCoefficients
    for indx2 = 2:mdl.NumCoefficients
        % beta term, check to see if the model description between the two
        % rows have similar elements (the equal test). If so, then the gain
        % is beta of indx2.
        gain_temp = 1;
        for indx_terms = 1:num_regr
            % Check to make sure that each term in the model is less than
            % the term (i.e., a subset).
            if mdl.model_desc(indx2,indx_terms) < mdl.model_desc(indx,indx_terms)
                gain_temp = 0;
            end
        end
        gain = gain_temp*mdl_temp.beta(indx2);

        % Check to see if the vectors are the same.
        if (mdl.model_desc(indx,:) == mdl.model_desc(indx2,:))
            gain2 = 1./x2fx(x_var_gain,mdl.model_desc(indx2,:));
        else
            gain2 = x2fx(x_var_offset,mdl.model_desc(indx,:) ˜= mdl.model_desc(indx2,:))./ ...
                    x2fx(x_var_gain,mdl.model_desc(indx2,:));
        end
        gamma(indx) = gamma(indx) + (gain.*gain2);
    end
end

% Create CoefficientNames Array
for indx = 1:mdl.NumCoefficients
    vars_used = find(mdl.model_desc(indx,:)>0);
    if ˜isempty(vars_used)
        if mdl.model_desc(indx,vars_used(1)) > 1
            mdl.CoefficientNames{indx} = [ds.Properties.VarNames{vars_used(1)} 'ˆ' num2str(mdl.model_desc(indx,vars_used(1)))];
        else
            mdl.CoefficientNames{indx} = ds.Properties.VarNames{vars_used(1)};
        end
        for indx2 = 2:length(vars_used)
            if mdl.model_desc(indx,vars_used(indx2)) > 1
                mdl.CoefficientNames{indx} = ...
                [mdl.CoefficientNames{indx} ':' ds.Properties.VarNames{vars_used(indx2)} 'ˆ' num2str(mdl.model_desc(indx,vars_used(indx2))];
            else
                mdl.CoefficientNames{indx} = ...
                [mdl.CoefficientNames{indx} ':',ds.Properties.VarNames{vars_used(indx2)}];
            end
        end
    end
end
mdl.beta = gamma;

E.4 State Variable Filtering

function [x_deriv, inp_filt] = calculate_derivative_alt(input, inp_time, alpha)
% This function calculates a derivative based on the equation
% x_deriv = s / (alpha s + 1) * input, and uses the bilinear transformation.
% inp_filt = 1/(alpha s + 1) * input.
% ts = mean(inp_time(2:end) - inp_time(1:end-1));
x_deriv(1) = (input(2)-input(1))/ts;
inp_filt(1) = input(1);
divisor = 2*alpha;
 País(1) = 0;
for indx = 1:length(input)-1
    x_deriv(indx+1) = (input(indx+2)-input(indx))/ts;
    inp_filt(indx+1) = input(indx);
end
\[ x_{\text{deriv}}(\text{indx}+1) = 2 \cdot \frac{\text{input}(\text{indx}+1)}{\text{divisor}} - 2 \cdot \frac{\text{input}(\text{indx})}{\text{divisor}} - \ldots \]
\[ x_{\text{deriv}}(\text{indx}) \cdot \frac{(t_s-2 \cdot \alpha)}{\text{divisor}}; \]
\[ \text{inp}_\text{filt}(\text{indx}+1) = t_s \cdot \frac{\text{input}(\text{indx}+1)+\text{input}(\text{indx})}{\text{divisor}} - \ldots \]
\[ \text{inp}_\text{filt}(\text{indx}) \cdot \frac{(t_s-2 \cdot \alpha)}{\text{divisor}}; \]
end
APPENDIX F

DISTANCE CALCULATION SOURCE CODE AND TEST

The code in Appendix F.1 is a test program to ensure the algorithm for the data set distance (see Section 4.1.2) has been implemented correctly. It creates three test vectors and then calculates the distance between the test vectors. See comments within the code for the expected results. The routine in Section F.2 reads a data file and then constructs the necessary vectors for the distance calculation. Then the code in Section F.3 performs the distance calculation.

F.1 Test Code for Calculation of Data Set Distance

clear all;
close all;
run ../set_path
add_path_data
% Creates test files for the distance calculation
% The model for delta 1 uncertainty is -500 + 100 * I_l - 1 * V_i;
% The model for delta 2 uncertainty is -1000 + 1000 * dt -> L = 1000.
% The file calc_data_set_distance is being tested.
% Validation data to tune data
% The scaled distance for the first uncertainty is sqrt((0.07*100)^2+15^2) = 16.5529
% The scaled distance for the second uncertainty is 500.
% validation to simulated validation data
% The scaled distance for the first uncertainty is sqrt(5.2^2 + 10^2) = 11.18
% The scaled distance for the second uncertainty is 300.
% Simulated Validation to tune data
% The scaled distance for the first uncertainty is 5.3852 between tune and sim_val sqrt(2^2 + 5^2).
% The scaled distance for the second uncertainty is 200.

% Create the Tuning data set
stime = [0:ts:1]';
logsout.PWM_cmd.Time = stime;
logsout.PWM_cmd.Data = rand(size(stime))*0.5;
logsout.Vil_data.Time = stime;
logsout.Vil_data.Data = rand(size(stime))*0.01)*0.5;
logsout.Vio_data.Time = stime;
logsout.Vio_data.Data = stime; % Placeholder
logsout.Vo_data.Time = stime;
F.2 Calculate Data Set Distance

% This routine is setup to calculate distance between two data sets
% (repeats it 3 times).
% Distance is defined as the maximum across all data points of set 2 the
% minimum distance between a point in data set 2 and all the data points in
% data set 1. Distance is a euclidean distance. As such it is order specific.
% function [distances,lipschitz_constant,Vo_vector_array,Il_vector_array] = ...
% calc_data_set_distance(PathFile_Tuning, PathFile_SimVal, PathFile_Val, ...
% pathName,delta_1_uncertainty_fcn,delta_2_uncertainty_fcn,TimeInt)
% Currently setup to load data from three files, Tuning data, Validation
% data, and simulation data.
% Returns distances in the order of :
% sim validation data to tune
% validation to tune data
% validation to sim validation data
% First is the voltage then the current.
%addpath('..\..\Statistics_Toolbox');
%addpath('..\..\Simulink_Library');
% This is the location of the data files and the functions
% delta_1_uncertainty_fcn and delta_2_uncertainty_fcn.
%add_path_data
Common_Resistors;
Plant.dT_Time_Forward = 0;
if (nargin < 1)
default_directory = 'C:\Users\zumberjt\Desktop\Jons_PhD_Data_Local\Reports\';
else
    default_directory = varargin{nargin};
end
% Load data files
[FileNames,PathName]=uigetfile({'*.mat'}, ...
'Select mat file(s) used for tuning','MultiSelect','on',default_directory);
if isempty(FileNames),
    return;
else
    for i=1:numel(FileNames),
        MATfiles{1,i}=(fullfile(PathName,FileNames{i}));
    end
end
[\',num_tests(1)] = size(FileNames);
[FileNames_pred,PathName_pred]=uigetfile({'*.mat'}, ...
'Select mat file(s) used for validation','MultiSelect','on',PathName);
if isempty(FileNames_pred),
    return;
else
    for i=1:numel(FileNames_pred),
        MATfiles{2,i}=(fullfile(PathName_pred,FileNames_pred{i}));
    end
end
else
    for i=1:numel(FileNames_pred),
        MATfiles{2,i}=fullfile(PathName_pred,FileNames_pred{i});
    end
end
[~,num_tests(2)] = size(FileNames_pred);

[FileNames_pred2,PathName_pred2]=uigetfile({'*.mat'},
    'Select set of mat file(s) used for simulation validation','MultiSelect','on',PathName_pred);
if isempty(FileNames_pred2),
    return;
end
if ˜iscell(FileNames_pred),
    MATfiles{3,1}=(fullfile(PathName_pred2,FileNames_pred2));
else
    for i=1:numel(FileNames_pred2),
        MATfiles{3,i}=fullfile(PathName_pred2,FileNames_pred2{i});
    end
end
[~,num_tests(3)] = size(FileNames_pred2);

delta_1_uncertainty_fcn = @Delta_1_Tuning_Model_Uncertainty;
%delta_2_uncertainty_fcn = @Delta_2_Tuning_Model_Uncertainty;
delta_2_uncertainty_fcn = @Delta_2_Tuning_Model_Uncertainty_debug;

TimeInt = [0.05 0.11];
pathName = pwd;
addpath(pathName)
else
    addpath(pathName);
end
[~,num_tests(1)] = size(PathFile_Tuning);
[~,num_tests(2)] = size(PathFile_Val);
[~,num_tests(3)] = size(PathFile_SimVal);
if ˜iscell(PathFile_Tuning),
    MATfiles{1,1}=(fullfile(pathName,PathFile_Tuning));
else
    for i=1:num_tests(1),
        MATfiles{1,i}=fullfile(pathName,PathFile_Tuning{i});
    end
end
if ˜iscell(PathFile_Val),
    MATfiles{2,1}=(fullfile(pathName,PathFile_Val));
else
    for i=1:num_tests(2),
        MATfiles{2,i}=fullfile(pathName,PathFile_Val{i});
    end
end
if ˜iscell(PathFile_SimVal),
    MATfiles{3,1}=(fullfile(pathName,PathFile_SimVal));
else
    for i=1:num_tests(3),
        MATfiles{3,i}=fullfile(pathName,PathFile_SimVal{i});
    end
end

FreqOfInterest = 5000; % Hz
AlphaOfInterest = 1/2/pi/FreqOfInterest;
FilterFreq = 10000; % Hz

for indx_fileset = 1:3 % Three different data sets.
    Vo_vector_final = [];
    Il_vector_final = [];
    Vc_vector_final = [];
    for indx1 = 1:num_tests(indx_fileset)
        % Load data files
        Temp_Tune = load(MATfiles{ indx_fileset, indx1 });
        if isfield(Temp_Tune,'logsout')
            % The following will force it to be a structure and thus be able to
            % examine whether certain signals are a field.
            SimulationDataUsed = 1;
            Datalog_Tune = struct(Temp_Tune.logsout);
        elseif isfield(Temp_Tune,'TDMS_Data')
            SimulationDataUsed = 0;
            Datalog_Tune = Temp_Tune.TDMS_Data;
        else
            error('Data Type not recognized');
        end
        % Create Time Vector that can be used by all.
        DataTime = Datalog_Tune.Vo_data.Time - Datalog_Tune.Vo_data.Time(1);
        % Rename variables

    end
end

188
Io = Datalog_Tune.Vio_data.Data/Plant.r_sns2;
Il = Datalog_Tune.Vil_data.Data/Plant.r_sns1;
Vo = Datalog_Tune.Vo_data.Data;

idxdT = find(DataTime>=Plant.dT_Time_Forward);
end_padding = numel(Vo) - numel(idxDT);
dT = [Datalog_Tune.PWM_cmd.Data(idxDT); ones(end_padding,1).*Datalog_Tune.PWM_cmd.Data(end)];

Vl = Datalog_Tune.Vl_data.Data;

% Calculate Derivative of data
IoDotFilt_Temp = calculate_derivative_alt(Io,DataTime,AlphaOfInterest)';
IoDotDotFilt_Temp = calculate_derivative_alt(IoDotFilt_Temp,DataTime,AlphaOfInterest)';

idx = find(DataTime>=TimeInt(1, 1) & DataTime<=TimeInt(1, 2));
%idx = find((DataTime>=TimeInt(1, 1) & DataTime<=TimeInt(1, 2)) | ...
% (DataTime>=TimeInt(2, 1) & DataTime<=TimeInt(2, 2)));

[~,Temp_vector] = delta_1_uncertainty_fcn(Datalog_Tune.PWM_cmd.Data',Il', ...
Datalog_Tune.Vo_data.Data',Datalog_Tune.Vio_data.Data'/Plant.r_sns2,Datalog_Tune.Vi_data.Data', ...
IoDotFilt_Temp',IoDotDotFilt_Temp');
Vo_vector = Temp_vector(2:end,:);

% If the delta_1_uncertainty_fcn returns just a constant then the
% Vo_vector will be empty. Need to check for it. If there is a
% constant, there is logic below that will set the distance and
% other return values to the correct values.
if isempty(Vo_vector)
Vo_vector_final = nan;
else
Vo_vector_final = [Vo_vector_final;Vo_vector(idx,:)];
end

[~,Temp_vector] = delta_2_uncertainty_fcn(Datalog_Tune.PWM_cmd.Data',Il', ...
Datalog_Tune.Vo_data.Data',Datalog_Tune.Vio_data.Data'/Plant.r_sns2,Datalog_Tune.Vi_data.Data', ...
IoDotFilt_Temp',IoDotDotFilt_Temp');
Il_vector = Temp_vector(2:end,:);

% If the delta_2_uncertainty_fcn returns just a constant then the
% vo_vector will be empty. Need to check for it. If there is a
% constant, there is logic below that will set the distance and
% other return values to the correct values.
if isempty(Il_vector)
Il_vector_final = nan;
else
Il_vector_final = [Il_vector_final;Il_vector(idx,:)];
end

Vo_vector_array{indx_fileset} = Vo_vector_final;
Il_vector_array{indx_fileset} = Il_vector_final;
end

Vo_dot = calculate_derivative_alt(Datalog_Tune.Vio_data.Data/Plant.r_sns2,Datalog_Tune.Vio_data.Time,AlphaOfInterest)';
Vo_dot_dot = calculate_derivative_alt(Vo_dot,Datalog_Tune.Vio_data.Time,AlphaOfInterest)';

[~,temp_GV,~] = delta_1_uncertainty_fcn(Datalog_Tune.PWM_cmd.Data',Il', ...
Datalog_Tune.Vo_data.Data',Datalog_Tune.Vio_data.Data'/Plant.r_sns2,Datalog_Tune.Vi_data.Data', ...
Vo_dot,Vo_dot_dot');

% Don't use the offset for the Lipschitz gain scaling (so skip the first
% element).
if numel(temp_GV) > 1
gain_vector_Vo = temp_GV(2:end)';
lipschitz_constant(1) = sqrt(numel(gain_vector_Vo));
% Between the validation and tune (attempt to find maximum distance from
% any validation data to a tuning data point):
D_eqv_val_tune_V = max(data_set_distance_2(Vo_vector_array{2},Vo_vector_array{1},gain_vector_Vo));

% Between the validation and sim validation (attempt to find maximum distance from
% any validation data to a simulated validation data point or any simulated validation to validation):
D_eqv_val_sim_val_V = max(data_set_distance_2(Vo_vector_array{2},Vo_vector_array{3},gain_vector_Vo));

% Between the simulated validation and tune (attempt to find maximum distance from
% any simulated validation data to a tuning data point):
D_eqv_sim_tune_V = max(data_set_distance_2(Vo_vector_array{3},Vo_vector_array{1},gain_vector_Vo));
else
% Only a constant in the uncertainty calculation.
gain_vector_Vo = 0;
lipschitz_constant(1) = 0;

end
\[ D_{\text{equ val tune V}} = 0; \]
\[ D_{\text{equ val sim V}} = 0; \]
\[ D_{\text{equ sim tune V}} = 0; \]
\end

\[ [\_\_\_, \text{temp GV}, \_\_\_\_] = \text{delta_2_uncertainty_fcn(Datalog T
\text{une}.PWM_cmd.Data', Il', ...
\text{Datalog T
\text{une}.Vo_data.Data', Datalog T
\text{une}.Vio_data.Data', Datalog T
\text{une}.V_i_data.Data', ... 
\text{IoDotFilt'}, \text{IoDotDotFilt'}); \]

\text{if (numel(temp GV) > 1)}
\text{gain vector}_\text{Il} = \text{temp GV}(2:end)';
\text{lipschitz constant}(2) = \text{sqrt(numel(gain vector}_\text{Il});
\text{\% Between the validation and tune (attempt to find maximum distance from \%
\text{any validation data to a tuning data point).}
\text{D}_{\text{equ val tune I}} = \max(\text{data set distance}_2(\text{Il vector array}_2, \text{Il vector array}_1, \text{gain vector}_\text{Il});
\text{\% Between the validation and sim validation (attempt to find maximum distance from \%
\text{any validation data to a simulated validation data point or any simulated validation to validation).}
\text{D}_{\text{equ val sim I}} = \max(\text{data set distance}_2(\text{Il vector array}_2, \text{Il vector array}_3, \text{gain vector}_\text{Il});
\text{\% Between the simulated validation and tune (attempt to find maximum distance from \%
\text{any simulated validation data to a tuning data point).}
\text{D}_{\text{equ sim tune I}} = \max(\text{data set distance}_2(\text{Il vector array}_3, \text{Il vector array}_1, \text{gain vector}_\text{Il});
\text{\% Between the validation and sim validation (attempt to find maximum distance from \%
\text{any validation data to a simulated validation data point or any simulated validation to validation).}
\text{D}_{\text{equ val sim I}} = \max(\text{data set distance}_2(\text{Il vector array}_2, \text{Il vector array}_3, \text{gain vector}_\text{Il});
\text{else}
\text{gain vector}_\text{Il} = 0;
\text{lipschitz constant}(2) = 0;
\text{D}_{\text{equ val tune I}} = 0;
\text{D}_{\text{equ val sim I}} = 0;
\text{D}_{\text{equ sim tune I}} = 0;
\text{\end}
\]

\text{disp('Compare validation to tune data (Vo dot): ' num2str(D_{\text{equ val tune V}}));}
\text{disp('Compare validation to tune data (Il dot): ' num2str(D_{\text{equ val tune I}}));}
\text{disp('Compare validation to simulated data (Vo dot): ' num2str(D_{\text{equ val sim V}}));}
\text{disp('Compare validation to simulated data (Il dot): ' num2str(D_{\text{equ val sim I}}));}
\text{disp('Compare simulated to tune data (Vo dot): ' num2str(D_{\text{equ sim tune V}}));}
\text{disp('Compare simulated to tune data (Il dot): ' num2str(D_{\text{equ sim tune I}}));}
\text{distances = [D_{\text{equ sim tune V}} D_{\text{equ val tune V}} D_{\text{equ val sim V}}; ...}
\text{D_{\text{equ sim tune I}} D_{\text{equ val tune I}} D_{\text{equ val sim I}}];}
\text{\text{rmpath(pathName)};

\text{F.3 Data Set Distance}

\text{function [ min distance ] = data set distance}_2(\text{x1, x2, gain vector })
\text{\% Euclidean distance between points}
\text{\% want to calculate min distance between a point in x1 and all points}
\text{\% in x2}
\text{\% Assumes each row is an observation, and each column is a different
\% dimension.}
\text{\% This version does not normalize the data.}
\text{\% Determine size of vectors}
\text{[x1_row, x1_col] = size(x1);}
\text{[x2_row, x2_col] = size(x2);}
\text{if [x1_col == x2_col]}
\text{error('Number of columns need to be the same.'));
\text{\end}
\text{\% No Normalization}
\text{x1_coded = coded_var(x1);}
\text{x2_coded = x2.*ones(x2_row,1).*gain vector';}
\text{\% Initialize the distance vector}
\text{min_distance = nan(x1_row,1);}
\text{\% for indx1 = 1:x1_row}
\text{x1_temp = repmat(x1_coded(indx1,:),x2_row,1);}
\text{min_distance(indx1) = \text{sqrt(min(dot((x1_temp-x2_coded),(x1_temp-x2_coded),2));}}
\text{\end
\text{end

190
APPENDIX G

UNIFORM ULTIMATE BOUNDARY CALCULATION AND CODE

G.1 Description

Based on the Lyapunov stability analysis in Section 5.3.1, the following is code for generating the theoretical invariant set used in this dissertation.

G.2 Code

% Calculate Maximum Boundary Conditions
% V = 1/2 e_1'2 + 1/2 (beta e_2) '2
% Should make this a function of W1 and W2.
clear all;
%close all;
add_path_data
%Boost_Circuit_Parameters_Setup_Single_C;
%BC_Parameters_Setup_Single_added_physics_fmin_deriv_v3
%BCP_14kHz_dTmod_fmin_3
%BCP_14kHz_dTmod_fmin_6_multiple_r;
BCP_Setup_Single_r;
Nonlinear_Damping_Analysis = 0;
if (Nonlinear_Damping_Analysis)
Controller.beta_val = 2.0452;
Controller.k1 = 1.0e+003*Gain;
Controller.k2 = 1.8703e+003*Gain;
Controller.c1 = 364.9591e-003*Gain*c_gain;
Controller.c2 = 0.1*Gain*c_gain;
Controller.c3 = 25*Gain*c_gain;
else
Controller.beta_val = 15+1+0;
Controller.k1 = 2000+1;
Controller.k2 = 10000+1;
Controller.c1 = 0;
Controller.c2 = 0;
Controller.c3 = 0;
end
% Controller.W1 = 150;
% Controller.W2 = 300;
% Controller.W3 = 0.05;
% Current values for BOC_Setup_Single_r (based on Lipschitz estimates
Controller.W1 = 84.42+60.08*1;
Controller.W2 = 106.27+274.33*1;
Controller.W3 = 0.22;

% Values needed for 200 mV tracking
% Controller.W1 = 12;
% Controller.W2 = 150;
% Controller.W3 = 0.22;

%% Calculate W3
% I_o = [0.045 0.045 0.045 0.045 0.09 0.09 0.09 0.09];
% V_i = [6 6 8 8 6 6 8 8];
% V_o = [9.2 8.8 9.2 8.8 9.2 8.8 9.2 8.8];
% V_i = [6 6 8 8 6 6 8 8]/2;
% V_o = [9.5 4.5 9.5 4.5 9.5 4.5 9.5 4.5]/2;
% I_o_bound(1:4) = V_o(1:4) / 200;
% I_o_bound(5:8) = V_o(5:8) / 100;
% V_o_ref = V_o;
%
% For backstepping control only.
% W3_temp = I_o ./ (V_i .* Controller.eta) - 2*Controller.k1*V_o*Controller.C ./ ( V_i*Controller.eta) + ...
% Controller.C * Controller.k1 * V_o_ref ./ ( V_i*Controller.eta);
% This is for nonlinear damping control and backstepping if the cl, c2, c3 = 0.
% W3_temp = I_o_bound ./ ( V_i*Controller.eta) - 2*Controller.k1*V_o*Controller.C ./ ( V_i*Controller.eta) + ...
% Controller.C * Controller.k1 * V_o_ref ./ ( V_i*Controller.eta) - ...
% 2 * Controller.C * Controller.c1 * Controller.W1ˆ2 ./ ( V_i*Controller.eta) .* V_o + ...
% Controller.C * Controller.c1 * Controller.W1ˆ2 * V_o_ref ./ ( V_i*Controller.eta);
W3 = abs(max(W3_temp));
if W3 > Controller.W3
  disp('Need to set W3 to at least');
  disp(W3);
end

%% Calculate Boundary Conditions
% k2_mod = Controller.k2 / 10ˆ2;
if (Controller.k2 > Controller.k1)
  k2_offset = Controller.k2 - Controller.k1;
  if ((Controller.k2-k2_offset) < 0)
    warning('Choose a different value for k2_offset');
  end
else
  k2_offset = 1;
end
k_v = min(Controller.k1-1,Controller.k2-k2_offset);
c1 = Controller.c1;
c2 = Controller.c2;
c3 = Controller.c3;
if (Nonlinear_Damping_Analysis)
  Wd = 1/(4*c1) + Controller.beta_val^2/(4*c2) + Controller.beta_val^2/(4*c3);
else
  % For backstepping only:
  Wd = Controller.W1^2 / 4 + Controller.beta_val^2*(Controller.W2 + W3 * Controller.W1).^2/(4*k2_offset);
end
Estimated_Voltage_Error = sqrt(2*Wd/k_v);
disp(['Estimated Voltage Error = ' num2str(Estimated_Voltage_Error)]);
Estimated_Current_Error = sqrt(2/Controller.beta_val^2*Wd/k_v);
disp(['Estimated Current Error = ' num2str(Estimated_Current_Error)]);

%% Draw Region of attraction based on Lyapunov
% |V| = wd / k_v

% Values from optimization
volt_rng = 4.0:0.02:9.0;
curr_rng = 0.0:0.01:1.0;
volt_cmd = 7.0;
...
% Varies the load resistance, the input voltage, the voltage command, and
% the inductor current to examine the Lyapunov region of attraction.
for R_L = 100:25:200
    f_1_nom = -volt_cmd/R_L / Plant.C;
    for indx_vin = 1:length(vin_rng)
        Vin = vin_rng(indx_vin);
        g_1_nom = Vin*nu/Controller.C/R_L;
        alpha_target = 1/g_1_nom - f_1_nom;
        for indx_volts = 1:length(volt_rng)
            for indx_current = 1:length(curr_rng)
                I_o = volt_rng(indx_volts) / R_L;
                [h_1, j_1, ~, ~, ~, ~] = ...
                    Delta_1_Regression_Control(nan,curr_rng(indx_current),volt_rng(indx_volts), ...
                    I_o,Vin,0,...
                    g_0,0);
                g_1 = Vin*nu/Controller.C/h_1;
                f_1 = -I_o/Controller.C + h_1;
                e_1(k) = volt_rng(indx_volts) - volt_cmd;
                alpha(k) = 1/g_1 *(-f_1 - Controller.k1 * e_1(k) - c1*Controller.W1^2*e_1(k));
                e_2(k) = curr_rng(indx_current) - alpha(k);
                V_plot(k) = 0.5*e_1(k)^2 + 0.5*e_2(k)^2*Controller.beta_val^2;
                if V_plot(k) < (Wd/k_v)
                    volt_plot_good(indx_g) = volt_rng(indx_volts);
                    current_plot_good(indx_g) = curr_rng(indx_current);
                    error_plot_good_volt(indx_g) = e_1(k);
                    error_plot_good_curr(indx_g) = e_2(k);
                    alpha_good(indx_g) = alpha(k);
                    indx_g = indx_g + 1;
                else
                    volt_plot_bad(indx_b) = volt_rng(indx_volts);
                    current_plot_bad(indx_b) = curr_rng(indx_current);
                    error_plot_bad_volt(indx_b) = e_1(k);
                    error_plot_bad_curr(indx_b) = e_2(k);
                end
                k = k+1;
            end
        end
    end
k2_eff = Controller.k2 + Controller.c2*Controller.W2^2 + Controller.c3*(Controller.W3*Controller.W1)^2;
k1_eff = Controller.k1 + Controller.c1*Controller.W1^2;
%costvalue = 1/(max(min(current_plot_good),1e-5))
ocstvalue = indx_g;
figure;
plot(volt_plot_bad_current_plot_bad,'r+', ...
     volt_plot_good_current_plot_good,'g+', ...
     volt_cmd,alpha_target,'k+');
xlabel('Voltage (V)');
ylabel('Current (A)');
title(['\beta val = ' num2str(Controller.beta_val)]);
figure('Name','Error States');
plot(error_plot_bad_volt,error_plot_bad_curr,'r+', ...
     error_plot_good_volt,error_plot_good_curr,'g+');
xlabel('e_1');
ylabel('e_2');
figure;
plot(e_1,e_2,'r+');
xlabel('e_1');
ylabel('e_2');

figure;
plot(e_1, alpha, '+');
xlabel('Voltage Error');
ylabel('alpha');

% calculate_boundary_conditions_fcn(x0, Controller)
end
APPENDIX H

UNCERTAINTY ANALYSIS CODE

H.1 Output Current and Inductor Current Uncertainty Code

The following is the m-code for implementation of the equations in Section 7.3.2.

```matlab
% These are the sense resistors. The means here are based on those items % that are in board revision 2 board number 2. % R_sns1 -> R32, R_sns2 -> R34 % The standard deviations are based on resistors 14 and 44
clear all;

r1_mean = 0.4701;
r1_std = 0.0008123909;
S_Vr1 = 4.89e-5; % (V) 95 % confidence random uncertainty due to the DAQ measurement with 35 Tests - this is the avg of channel 3 and 4

% r_sns2
r2_mean = 0.469;
r2_std = 0.0008123909; % with 50 tests
S_Vr2 = 4.89e-5; % (V) 95 % confidence random uncertainty due to the DAQ measurement with 35 Tests - this is the avg of channel 3 and 4

% Fluke systematic uncertainty in voltage voltage range of +/- 1 V.
B_Vr = 0.000011766; % Same systematic error for the two channels.
N_r = 50;
N_v = 35;

if (1)
    % To setup calculation for r_sns2;
    r_mean = r2_mean;
r_std = r2_std;
    S_Vr = S_Vr2;
v_mean = 0.090*r_mean; % 90mA is the worst case scenario. of the output current;
else
    % For calculation of r_sns1;
    r_mean = r1_mean;
r_std = r1_std;
    S_Vr = S_Vr1;
v_mean = 0.210*r_mean; % 90mA is the worst case scenario. of the output current;
end

% LCR systematic uncertainty +/- 0.98% at 200 Hz.
B_R = 0.0098;

% Resistance Uncertainty.
b_r = B_R/2*r_mean; % Systematic Uncertainty
s_r = r_std / sqrt(1 + 1/N_r); % Random Uncertainty

% Voltage Uncertainty (Systematic) r_sns2 is on channel AI3.
b_v = B_Vr/2; % Systematic Uncertainty
s_v = S_Vr/2*sqrt(1 + 1/N_v); % Random Uncertainty

% Partial Calculations
partial_I_wrt_V = 1/r_mean;
partial_I_wrt_R = v_mean/r_mean^2;
s_t = sqrt((b_r*partial_I_wrt_R)^2 + (b_v*partial_I_wrt_V)^2);
b_t = sqrt((b_r*partial_I_wrt_R)^2 + (b_v*partial_I_wrt_V)^2);
```

195
\[ \text{deg_of_freedom} = \left( \frac{\partial I_wrt V \cdot s_v}{(N_v - 1)} + \frac{\partial I_wrt V \cdot b_v}{(N_v - 1)} \right)^2 + \left( \frac{\partial I_wrt R \cdot s_r}{(N_r - 1)} + \frac{\partial I_wrt R \cdot b_r}{(N_r - 1)} \right)^2 \]

\[ t_{\text{gain}} = tinv(1 - 0.05/2, \text{deg_of_freedom}); \]

This calculation is the prediction interval on the current measurement.

\[ u_{\text{final}} = t_{\text{gain}} \left( s_t^2 + b_t^2 \right)^{0.5}; \]

### H.2 Derivative Uncertainty Code

The following code was developed for the analysis in Section 7.3.3.

```matlab
%% This routine will aid in assessing the uncertainty in the derivative calculation.
clear all;close all;

% Voltage measurement uncertainty (95% conf) / Current measurement uncertainty
% variance of the voltage measurement.
FREQOfInterest= 5000;  % Hz
AlphaOfInterest = 1/2/pi/FREQOfInterest;
test_freq = 5000;
time = 1/14e3;
stime_up_sample = 0:time:0.2;
sig_1_up_sample = 2*sin(test_freq*2*pi*stime_up_sample) + 5;
%upsample
sig_1_up_sample_filtered = filter(B_10k_disc,A_10k_disc,sig_1_up_sample);

% Use the calculate_derivative_alt due to the filtering.
sig_1_deriv = 2*test_freq*2*pi*cos(test_freq*2*pi*stime);
sig_1_deriv_est = calculate_derivative_alt(sig_1,stime,AlphaOfInterest);
noise_est = nan(5000,1);
noise_level = nan(5000,1);
for indx = 1:5000
    sig_1_w_noise = sig_1_up_sample + randn(size(stime_up_sample))*sqrt(v_var);
    sig_1_w_noise_up_sample_filtered = filter(B_10k_disc,A_10k_disc,sig_1_w_noise_up_sample);
    sig_1_w_noise = downsample(sig_1_w_noise_up_sample_filtered,2);
    sig_1_w_noise_derivative_est = calculate_derivative_alt(sig_1_w_noise,stime,AlphaOfInterest);
    noise_est(indx) = std(sig_1_w_noise_derivative_est-sig_1_deriv);
    noise_level(indx) = abs(mean(sig_1_w_noise_derivative_est-sig_1_deriv)) + 2*noise_est(indx);
end

% This is the value reported in my dissertation (just use the average of the above numbers.
mean(noise_level)
figure;
plot(stime,sig_1_deriv,"+",stime,sig_1_deriv_est,'x');
legend('Analytical Derivative','Derivative Estimate');

figure('Name','Noise');
plot(stime,sig_1_deriv-sig_1_deriv_est);
```

196
APPENDIX I

RESULTS OF PARAMETER SET FROM FMINCON WITH $J_{\bar{z}}$ COST FUNCTION

The following appendix presents information from the parameter set created with the optimization function fmincon using the cost function $J_{\bar{z}}$.

I.1 Model Uncertainty Analysis with Data Set $D_t$

The following are the polynomials created for this model (See Tables I.1 and I.3) along with the regression diagnostic tables (see Tables I.2 and I.4).

Table I.1: $\delta_{m1}$, Polynomial for Data Set $D_t$ for Model Fmincon w $J_{\bar{z}}$.

<table>
<thead>
<tr>
<th></th>
<th>Estimate</th>
<th>SE</th>
<th>tStat</th>
<th>pValue</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Intercept)</td>
<td>67.8</td>
<td>0.388</td>
<td>175</td>
<td>0</td>
</tr>
<tr>
<td>$V_i$</td>
<td>-7.33</td>
<td>0.0869</td>
<td>-84.3</td>
<td>0</td>
</tr>
<tr>
<td>$I_o$</td>
<td>-977</td>
<td>6.59</td>
<td>-148</td>
<td>0</td>
</tr>
<tr>
<td>$d_T$</td>
<td>-117</td>
<td>0.466</td>
<td>-250</td>
<td>0</td>
</tr>
<tr>
<td>$I_o : d_T$</td>
<td>2.49e+03</td>
<td>10</td>
<td>248</td>
<td>0</td>
</tr>
</tbody>
</table>

I.2 Model Uncertainty Analysis with Data Set $D_v$

The following are the model uncertainties based on data from data set $D_v$ and $D_t$. See Tables I.5 and I.7 for the polynomials. See Tables I.6 and I.8 for the regression diagnostics.
Table I.2: Regression Diagnostic Table for $\delta_{m1}$ for Data Set $D_t$ for Model Fmincon with $J_{\dot{z}}$ cost function.

<table>
<thead>
<tr>
<th>Std. Dev.</th>
<th>R-Squared</th>
<th>pValue</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.1272</td>
<td>0.8952</td>
<td></td>
</tr>
<tr>
<td>Mean</td>
<td>5.2127</td>
<td>0.8840</td>
</tr>
<tr>
<td>PRESS</td>
<td>707288.0483</td>
<td>0.8952</td>
</tr>
</tbody>
</table>

Table I.3: $\delta_{m2}$, Polynomial for Data Set $D_t$ for Model Fmincon with $J_{\dot{z}}$.

<table>
<thead>
<tr>
<th>Estimate</th>
<th>SE</th>
<th>tStat</th>
<th>pValue</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Intercept)</td>
<td>368</td>
<td>2.55</td>
<td>145</td>
</tr>
<tr>
<td>$d_T$</td>
<td>-808</td>
<td>6.41</td>
<td>-126</td>
</tr>
<tr>
<td>$V_i$</td>
<td>-50.7</td>
<td>0.514</td>
<td>-98.6</td>
</tr>
<tr>
<td>$d_T^2$</td>
<td>781</td>
<td>6.72</td>
<td>116</td>
</tr>
</tbody>
</table>

Table I.4: Regression Diagnostic Table for $\delta_{m2}$ for Data Set $D_t$ for Model Fmincon with $J_{\dot{z}}$.

<table>
<thead>
<tr>
<th>Std. Dev.</th>
<th>R-Squared</th>
<th>pValue</th>
</tr>
</thead>
<tbody>
<tr>
<td>32.5432</td>
<td>0.4532</td>
<td></td>
</tr>
<tr>
<td>Mean</td>
<td>6.8374</td>
<td>0.4154</td>
</tr>
<tr>
<td>PRESS</td>
<td>28494814.0725</td>
<td>0.4531</td>
</tr>
</tbody>
</table>

Table I.5: $\delta_{m1}$, Polynomial for Data Set $D_v$ for Model Fmincon with $J_{\dot{z}}$.

<table>
<thead>
<tr>
<th>Estimate</th>
<th>SE</th>
<th>tStat</th>
<th>pValue</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Intercept)</td>
<td>66.5</td>
<td>0.319</td>
<td>208</td>
</tr>
<tr>
<td>$V_i$</td>
<td>-9.3</td>
<td>0.065</td>
<td>-143</td>
</tr>
<tr>
<td>$I_o$</td>
<td>-730</td>
<td>5.03</td>
<td>-145</td>
</tr>
<tr>
<td>$d_T$</td>
<td>-108</td>
<td>0.399</td>
<td>-272</td>
</tr>
<tr>
<td>$I_o : d_T$</td>
<td>2.17e+03</td>
<td>8.27</td>
<td>263</td>
</tr>
</tbody>
</table>

Table I.6: Regression Diagnostic table for $\delta_{m1}$ for Model Fmincon with $J_{\dot{z}}$.

<table>
<thead>
<tr>
<th>Std. Dev.</th>
<th>R-Squared</th>
<th>pValue</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.4011</td>
<td>0.8243</td>
<td></td>
</tr>
<tr>
<td>Mean</td>
<td>1.1678</td>
<td>0.8192</td>
</tr>
<tr>
<td>PRESS</td>
<td>2649201.3263</td>
<td>0.8243</td>
</tr>
</tbody>
</table>

198
Table I.7: $\delta_m^2$ Polynomial for Data Set $D_v$ for Model Fmincon with $J_\dot{x}$.

<table>
<thead>
<tr>
<th>Estimate</th>
<th>SE</th>
<th>tStat</th>
<th>pValue</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intercept</td>
<td>322</td>
<td>1.43</td>
<td>225</td>
</tr>
<tr>
<td>$d_T$</td>
<td>-777</td>
<td>3.86</td>
<td>-201</td>
</tr>
<tr>
<td>$V_i$</td>
<td>-40.9</td>
<td>0.288</td>
<td>-142</td>
</tr>
<tr>
<td>$d_T^2$</td>
<td>775</td>
<td>4.06</td>
<td>191</td>
</tr>
</tbody>
</table>

Table I.8: Regression Diagnostic Table for $\delta_m^2$ for Model Fmincon with $J_\dot{x}$.

| Std. Dev. | 25.3230 | R-Squared | 0.4252 |
| Mean      | -5.6565 | R-Squared Adj | 0.4143 |
| PRESS     | 58235585.3774 | R-Squared Pred | 0.4252 |

I.3 Augmented Controller Construction

The following is the polynomial used in the augmented controller (see Table I.9) along with the regression diagnostics (see Table I.10).

Table I.9: $\delta_c^1$ Polynomial for Data Set $D_l$ for Model Fmincon with $J_\dot{x}$.

<table>
<thead>
<tr>
<th>Estimate</th>
<th>SE</th>
<th>tStat</th>
<th>pValue</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intercept</td>
<td>17.8</td>
<td>0.368</td>
<td>48.3</td>
</tr>
<tr>
<td>$I_l$</td>
<td>-570</td>
<td>2.4</td>
<td>-237</td>
</tr>
<tr>
<td>$V_i$</td>
<td>-8.68</td>
<td>0.0953</td>
<td>-91.2</td>
</tr>
<tr>
<td>$I_o$</td>
<td>-576</td>
<td>3.35</td>
<td>-172</td>
</tr>
<tr>
<td>$I_l : V_i$</td>
<td>267</td>
<td>0.662</td>
<td>404</td>
</tr>
<tr>
<td>$I_l : I_o$</td>
<td>-7.54e+03</td>
<td>12.1</td>
<td>-623</td>
</tr>
</tbody>
</table>
Table I.10: Regression Diagnostic Table for $\delta_{c1}$ for Model Fmincon with $J_x$.

| Std. Dev. | 3.7849 | R-Squared | 0.9965 |
| Mean      | -67.8935 | R-Squared Adj | 0.9959 |
| PRESS     | 385406.0121 | R-Squared Pred | 0.9965 |

I.4 Control Uncertainty Analysis with Data Set $D_v$

The following is the control uncertainty polynomial construction using data set $D_v$ (see Table I.11). This model is similar to the fmincon with $J_x$ as the cost function (see Chapter IX). The regression diagnostics are in Table I.12.

Table I.11: $\delta_{c1}$ Polynomial for Data Set $D_v$ for Model Fmincon with $J_x$.

<table>
<thead>
<tr>
<th>Estimate</th>
<th>SE</th>
<th>tStat</th>
<th>pValue</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Intercept)</td>
<td>45.9</td>
<td>0.25</td>
<td>184</td>
</tr>
<tr>
<td>$I_l$</td>
<td>-446</td>
<td>3.34</td>
<td>-133</td>
</tr>
<tr>
<td>$I_o$</td>
<td>-686</td>
<td>4.76</td>
<td>-144</td>
</tr>
<tr>
<td>$\dot{I}_o$</td>
<td>-0.603</td>
<td>0.00531</td>
<td>-114</td>
</tr>
<tr>
<td>$I_l : I_o$</td>
<td>6.51e+03</td>
<td>41.6</td>
<td>157</td>
</tr>
</tbody>
</table>

Table I.12: Regression Diagnostic table for $\delta_{c1}$ for Data Set $D_v$ for Model Fmincon with $J_x$.

| Std. Dev. | 4.4485 | R-Squared | 0.4367 |
| Mean      | 2.2632 | R-Squared Adj | 0.4132 |
| PRESS     | 1264750.6088 | R-Squared Pred | 0.4366 |
I.5 Control Uncertainty Analysis with Data Set $D_s$

The following is the control uncertainty polynomial construction using data set $D_s$ (see Table I.13). This model is similar to the one constructed in the previous section, with a slightly higher coefficient of determination (see Table I.14).

Table I.13: $\delta_{c1}$ Polynomial for Data Set $D_s$ for Model Fmincon with $J_\dot{x}$.

<table>
<thead>
<tr>
<th></th>
<th>Estimate</th>
<th>SE</th>
<th>tStat</th>
<th>pValue</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Intercept)</td>
<td>38.5</td>
<td>0.237</td>
<td>162</td>
<td>0</td>
</tr>
<tr>
<td>$I_l$</td>
<td>-383</td>
<td>3.28</td>
<td>-117</td>
<td>0</td>
</tr>
<tr>
<td>$I_o$</td>
<td>-558</td>
<td>4.55</td>
<td>-122</td>
<td>0</td>
</tr>
<tr>
<td>$\dot{I}_o$</td>
<td>-0.671</td>
<td>0.00544</td>
<td>-123</td>
<td>0</td>
</tr>
<tr>
<td>$I_l : I_o$</td>
<td>5.33e+03</td>
<td>40.6</td>
<td>131</td>
<td>0</td>
</tr>
</tbody>
</table>

Table I.14: Regression Diagnostic Table for $\Delta_{c1}$ for Data Set $D_s$ for Model Fmincon with $J_\dot{x}$.

<table>
<thead>
<tr>
<th></th>
<th>Std. Dev.</th>
<th>R-Squared</th>
<th>R-Squared Adj</th>
<th>R-Squared Pred</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean</td>
<td>0.9853</td>
<td>0.4498</td>
<td>0.4268</td>
<td>0.4497</td>
</tr>
<tr>
<td>PRESS</td>
<td>1292177.0389</td>
<td>0.4497</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>