AN ABSTRACT OF THE THESIS OF

Christopher A. Haller for the degree of Master of Science in Electrical and Computer Engineering presented on June 10, 2010.

Title: Calibration, Characterization, and Linear Quadratic Gaussian Estimation of Sensor Feedback Signals for a Novel Ocean Wave Energy Linear Test Bed

Abstract approved: ________________________________________________

Ted K. A. Brekken

Continual growth of the world’s energy consumption and the demand for long-term clean renewable energy resources has led to the development of ocean-based wave energy technologies. Emerging technologies are advancing potential solutions to the complex problem of energy generation in the harsh and corrosive ocean environment. To develop these technologies, new tools are needed for simulation and analysis of the ocean’s interactions with newly designed wave energy generating devices. OSU’s Ocean Wave Energy Linear Test Bed is one such tool that is capable of testing and providing baseline simulations of ocean wave interactions with point absorber ocean wave energy generating devices.

To provide accurate ocean wave simulation with the Linear Test Bed, a closed-loop force control scheme must be developed. To date, this force control scheme has been unsuccessful because of ambient Gaussian noise present in the machine’s feedback
signal lines. A Linear Quadratic Gaussian Estimator was constructed to solve this noise issue by providing estimates of the Linear Test Bed’s feedback signals and system states. The Kalman filter is constructed from a mathematical model of the Linear Test Bed physical movements and through statistical analysis of the Gaussian noise present in the feedback signal lines. The concern of sensor accuracy is also addressed by a calibration check of the machine’s force measuring load cells and through development of a simple methodology of calibrating the load cells. The major results of this thesis show that Kalman filtration of the Linear Test Bed’s feedback signals will provide relatively noise-free estimates that do not suffer phase delay issues common to the low-pass filter, while Kalman estimations of feedback signals reduce noise as measured by signal variance. The load cell sensors and signal lines experience drift over time, and therefore must be calibrated to more accurately reflect the state of an ocean energy generating device under test. Development of this machine’s advanced control systems will allow quicker and more accurate testing of ocean-based energy generating devices, that can help solve the world’s growing energy needs.
Calibration, Characterization, and Linear Quadratic Gaussian Estimation of Sensor Feedback Signals for a Novel Ocean Wave Energy Linear Test Bed

by

Christopher A. Haller

A THESIS

submitted to

Oregon State University

in partial fulfillment of the requirements for the degree of

Master of Science

Presented June 10, 2010
Commencement June 2011

APPROVED:

__________________________
Major Professor, representing Electrical and Computer Engineering

__________________________
Director of the School of Electric Engineering and Computer Science

__________________________
Dean of the Graduate School

I understand that my thesis will become part of the permanent collection of Oregon State University libraries. My signature below authorizes release of my thesis to any reader upon request.

__________________________
Christopher A. Haller, Author
I would like to thank my primary advisor, Dr. Ted Brekken, for his excellent controls guidance, support, and enthusiasm throughout the work leading up to this thesis. Also to Dr. Annette von Jouanne, who helped foster my initial involvement in the Energy Systems group at OSU, and has always been inspirational and supportive of not only my, but all ocean energy research.

A special thanks goes to my committee members who without I could not complete the Master of Science process: my major advisor Dr. Ted Brekken, and major co-advisor Dr. von Jouanne, my controls instructor Dr. Matt Knudson, and my probability professor Dr. Mina Ossiander.

Many thanks to the people that helped me down the writing path with excellent editing assistance: Channa Newell, Douglas Halamay, and Bret Bosma.

Also thanks goes to my fellow WESRFians in the Energy Systems group, especially Bret who has always been willing to talk about control theory, and Ean, who helped with load cell calibration.

Finally, my greatest thanks goes to my family, Ted, Cheryl, Michael, Dell, Anna, and Channa; for their ongoing love and encouragement.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1    Introduction</td>
<td>1</td>
</tr>
<tr>
<td>1.1  Research Focus</td>
<td>2</td>
</tr>
<tr>
<td>1.1.1 Load Cell Calibration</td>
<td>2</td>
</tr>
<tr>
<td>1.1.2 Kalman Filtration of Feedback Signals</td>
<td>3</td>
</tr>
<tr>
<td>1.2  MSRF/WESRF</td>
<td>3</td>
</tr>
<tr>
<td>1.3  NNMREC</td>
<td>4</td>
</tr>
<tr>
<td>2    Wave Energy Linear Test Bed</td>
<td>6</td>
</tr>
<tr>
<td>2.1  Background</td>
<td>8</td>
</tr>
<tr>
<td>2.1.1 LTB Origin</td>
<td>8</td>
</tr>
<tr>
<td>2.1.2 Description of Motion</td>
<td>8</td>
</tr>
<tr>
<td>2.1.3 L-10 Generator</td>
<td>10</td>
</tr>
<tr>
<td>2.1.4 Past Research</td>
<td>11</td>
</tr>
<tr>
<td>2.2  External Hardware and Software</td>
<td>12</td>
</tr>
<tr>
<td>2.2.1 MATLAB and Simulink</td>
<td>13</td>
</tr>
<tr>
<td>2.2.2 Rapid Prototyping System - dSPACE DSP and ControlDesk</td>
<td>14</td>
</tr>
<tr>
<td>3    Load Cell Calibration</td>
<td>17</td>
</tr>
<tr>
<td>3.1  Load Cell Description</td>
<td>18</td>
</tr>
<tr>
<td>3.1.1 Internal Strain Gauge Sensor of Load Cell</td>
<td>19</td>
</tr>
<tr>
<td>3.2  Partial Calibration Check</td>
<td>23</td>
</tr>
<tr>
<td>3.2.1 Hardware Configuration</td>
<td>23</td>
</tr>
<tr>
<td>3.2.2 Left and Right Load Cell Measurements Taken</td>
<td>29</td>
</tr>
<tr>
<td>3.2.3 Error Analysis</td>
<td>31</td>
</tr>
<tr>
<td>3.2.4 Best Fit Corrections</td>
<td>35</td>
</tr>
<tr>
<td>3.3  Recommended Calibration Procedure</td>
<td>37</td>
</tr>
<tr>
<td>3.3.1 Known Weights</td>
<td>40</td>
</tr>
<tr>
<td>3.3.2 Elastic Calibration Device</td>
<td>41</td>
</tr>
<tr>
<td>3.3.3 Manufacturer Calibration Methodology</td>
<td>41</td>
</tr>
<tr>
<td>3.3.4 Dynamic Load Cell Testing</td>
<td>42</td>
</tr>
<tr>
<td>3.3.5 Calibration Location</td>
<td>43</td>
</tr>
<tr>
<td>4    The Kalman Filter</td>
<td>44</td>
</tr>
<tr>
<td>4.1  Linear Quadratic Gaussian Controller</td>
<td>45</td>
</tr>
</tbody>
</table>
# TABLE OF CONTENTS (Continued)

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.2</td>
<td>Background of the Kalman Filter</td>
<td>47</td>
</tr>
<tr>
<td>4.3</td>
<td>Process Noise</td>
<td>48</td>
</tr>
<tr>
<td>4.3.1</td>
<td>Process Noise Measurement for the Linear Test Bed</td>
<td>50</td>
</tr>
<tr>
<td>4.3.1.1</td>
<td>Noise Measurement Procedure</td>
<td>50</td>
</tr>
<tr>
<td>4.3.1.2</td>
<td>Covariance Results From Noise Data</td>
<td>51</td>
</tr>
<tr>
<td>4.3.1.3</td>
<td>Comparison of Covariance Matrices</td>
<td>54</td>
</tr>
<tr>
<td>4.4</td>
<td>Transfer Function Characterization of the Linear Test Bed</td>
<td>58</td>
</tr>
<tr>
<td>4.4.1</td>
<td>Velocity Transfer Function</td>
<td>58</td>
</tr>
<tr>
<td>4.4.1.1</td>
<td>Velocity Transfer Function - Alternate Derivation</td>
<td>60</td>
</tr>
<tr>
<td>4.4.2</td>
<td>Acceleration Transfer Function</td>
<td>62</td>
</tr>
<tr>
<td>4.4.2.1</td>
<td>Acceleration Transfer Function - Alternate Derivation</td>
<td>64</td>
</tr>
<tr>
<td>4.4.3</td>
<td>Position Transfer Function</td>
<td>65</td>
</tr>
<tr>
<td>4.4.3.1</td>
<td>System Response Data Capture</td>
<td>66</td>
</tr>
<tr>
<td>4.4.3.2</td>
<td>Bode Plot Construction</td>
<td>69</td>
</tr>
<tr>
<td>4.4.3.3</td>
<td>Fitting 1st Order Transfer Function to Bode Plot</td>
<td>74</td>
</tr>
<tr>
<td>4.4.3.4</td>
<td>Fitting 4th Order Transfer Function to Bode Plot</td>
<td>77</td>
</tr>
<tr>
<td>4.4.4</td>
<td>Force Transfer Function Characterization</td>
<td>80</td>
</tr>
<tr>
<td>4.4.4.1</td>
<td>Load Cell Resistive Simplification</td>
<td>81</td>
</tr>
<tr>
<td>4.4.4.2</td>
<td>Step Response Analysis</td>
<td>82</td>
</tr>
<tr>
<td>4.4.4.3</td>
<td>Transfer Function Fit to Step Response Results</td>
<td>87</td>
</tr>
<tr>
<td>4.5</td>
<td>Kalman Filter Assembly</td>
<td>92</td>
</tr>
<tr>
<td>4.5.1</td>
<td>Sub-System Inner-Connection</td>
<td>92</td>
</tr>
<tr>
<td>4.5.2</td>
<td>Tuning</td>
<td>94</td>
</tr>
<tr>
<td>4.5.2.1</td>
<td>Position Transfer Function Tuning - $\omega_P$</td>
<td>94</td>
</tr>
<tr>
<td>4.5.2.2</td>
<td>Velocity Transfer Function Tuning - $\omega_V$</td>
<td>95</td>
</tr>
<tr>
<td>4.5.2.3</td>
<td>Acceleration Transfer Function Tuning - $\omega_A$</td>
<td>95</td>
</tr>
<tr>
<td>4.5.2.4</td>
<td>Force Transfer Function Tuning - $Q_N$</td>
<td>96</td>
</tr>
<tr>
<td>4.5.3</td>
<td>LTB Kalman Filter Matrices</td>
<td>97</td>
</tr>
<tr>
<td>4.6</td>
<td>Kalman Filter Results</td>
<td>99</td>
</tr>
<tr>
<td>4.6.1</td>
<td>Steady State Error Analysis</td>
<td>99</td>
</tr>
<tr>
<td>4.6.2</td>
<td>Simulated Ocean Wave Data</td>
<td>102</td>
</tr>
<tr>
<td>4.6.2.1</td>
<td>Position</td>
<td>103</td>
</tr>
<tr>
<td>4.6.2.2</td>
<td>Velocity</td>
<td>105</td>
</tr>
<tr>
<td>4.6.2.3</td>
<td>Acceleration</td>
<td>106</td>
</tr>
<tr>
<td>4.6.2.4</td>
<td>Force</td>
<td>108</td>
</tr>
</tbody>
</table>
### TABLE OF CONTENTS (Continued)

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>5 Conclusion</td>
<td>111</td>
</tr>
<tr>
<td>5.1 Load Cell Conclusion</td>
<td>111</td>
</tr>
<tr>
<td>5.2 Kalman Filter Conclusion</td>
<td>112</td>
</tr>
<tr>
<td>5.3 Recommended Areas of Future Work</td>
<td>112</td>
</tr>
<tr>
<td>5.3.1 Force Control Completion</td>
<td>112</td>
</tr>
<tr>
<td>5.3.2 Acceleration Signal</td>
<td>112</td>
</tr>
<tr>
<td>5.4 Ocean Energy Conclusion</td>
<td>114</td>
</tr>
<tr>
<td>Bibliography</td>
<td>115</td>
</tr>
<tr>
<td>Appendices</td>
<td>118</td>
</tr>
</tbody>
</table>


# LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Wave Energy Linear Test Bed (LTB) Installed at WESRF with L-10 Ocean Wave Linear Generator</td>
<td>7</td>
</tr>
<tr>
<td></td>
<td>(10 kW) Mounted in LTB</td>
<td></td>
</tr>
<tr>
<td>2.2</td>
<td>Solidworks Model of Complete L-10 Wave Energy Generator[3]</td>
<td>10</td>
</tr>
<tr>
<td>2.3</td>
<td>Simulink Blockset for dSPACE DS1103 Rapid Prototyping System[4]</td>
<td>14</td>
</tr>
<tr>
<td>2.4</td>
<td>dSPACE DS1103 Rapid Prototyping System Internal Block Diagram[4]</td>
<td>15</td>
</tr>
<tr>
<td>2.5</td>
<td>ControlDesk Layout Example for Interface with dSPACE DSP[5]</td>
<td>16</td>
</tr>
<tr>
<td>3.1</td>
<td>Interface Force 1000 Series Load Cell [6]</td>
<td>18</td>
</tr>
<tr>
<td>3.2</td>
<td>LTB Load Cell Suspension Arm[7]</td>
<td>19</td>
</tr>
<tr>
<td>3.3</td>
<td>Wheatstone Bridge Inside Load Cell</td>
<td>20</td>
</tr>
<tr>
<td>3.4</td>
<td>Strain Gauge Resistance States for Load Cell Wheatstone Bridge</td>
<td>22</td>
</tr>
<tr>
<td>3.5</td>
<td>Load Cell Suspension Arm Gimbal Configuration for Tension Testing</td>
<td>24</td>
</tr>
<tr>
<td>3.6</td>
<td>Load Cell Suspension Arm Gimbal Configuration for Compression Testing</td>
<td>25</td>
</tr>
<tr>
<td>3.7</td>
<td>S-Type Load Cell and Associated Digital Readout Used in Compression Testing</td>
<td>27</td>
</tr>
<tr>
<td>3.8</td>
<td>Load Cell Suspension Arm Testing Left Image: Load Cell Compression with Gantry Crane Right Image: Load Cell Tension with 2000 lb Motor</td>
<td>28</td>
</tr>
<tr>
<td>3.9</td>
<td>Partial Calibration Check Load Cell Percentage Accuracy</td>
<td>32</td>
</tr>
<tr>
<td>3.10</td>
<td>Partial Cause of Right Load Cell Compression Test Inaccuracy</td>
<td>34</td>
</tr>
<tr>
<td>3.11</td>
<td>Measured vs. Actual Force Measurements for Left Load Cell</td>
<td>35</td>
</tr>
<tr>
<td>3.12</td>
<td>Measured vs. Actual Force Measurements for Right Load Cell</td>
<td>36</td>
</tr>
<tr>
<td>3.13</td>
<td>Compression and Tension with an Elastic Calibration Device</td>
<td>38</td>
</tr>
<tr>
<td>4.1</td>
<td>Process and Measurement Noise Introduced into System</td>
<td>44</td>
</tr>
<tr>
<td>4.2</td>
<td>Noisy Feedback Signals from LTB</td>
<td>45</td>
</tr>
<tr>
<td>Figure</td>
<td>Description</td>
<td>Page</td>
</tr>
<tr>
<td>--------</td>
<td>------------------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>4.3</td>
<td>Linear Quadratic Gaussian Controller and Plant</td>
<td>46</td>
</tr>
<tr>
<td>4.4</td>
<td>L.Q.G. Estimator (Kalman Filter) and LTB Signals to be Filtered</td>
<td>47</td>
</tr>
<tr>
<td>4.5</td>
<td>LTB Noise Measurements, 10,000 Samples, 1 Second Duration</td>
<td>51</td>
</tr>
<tr>
<td>4.6</td>
<td>LTB Position Signal Flow Block Diagram</td>
<td>66</td>
</tr>
<tr>
<td>4.7</td>
<td>Slowest Commanded Position, 10 Sec. Period, 0.025 M Amplitude</td>
<td>68</td>
</tr>
<tr>
<td>4.8</td>
<td>Fastest Commanded Position, 0.25 Sec. Period, 0.025 M Amplitude</td>
<td>69</td>
</tr>
<tr>
<td>4.9</td>
<td>Position Transfer Function Bode Plot, from Peak and Zero Crossing Comparisons</td>
<td>70</td>
</tr>
<tr>
<td>4.10</td>
<td>Position Transfer Function Bode Plot, from FFT Analysis of Sinusoids</td>
<td>72</td>
</tr>
<tr>
<td>4.11</td>
<td>Position Transfer Function Bode Plot, Combined Analysis Methods</td>
<td>73</td>
</tr>
<tr>
<td>4.12</td>
<td>Commanded to Reported Position Mean Squared Error Analysis of Bode Plot</td>
<td>75</td>
</tr>
<tr>
<td>4.13</td>
<td>Position Transfer Function Bode Plot with $\omega_c$ Values from 0.4 to 0.8</td>
<td>76</td>
</tr>
<tr>
<td>4.14</td>
<td>Position Transfer Function (4th order) Bode Plot from MATLAB System Identification Tool</td>
<td>79</td>
</tr>
<tr>
<td>4.15</td>
<td>LTB Force Signal Flow Block Diagram</td>
<td>81</td>
</tr>
<tr>
<td>4.16</td>
<td>Load Cell Step Response Signal Creation Block Diagram</td>
<td>84</td>
</tr>
<tr>
<td>4.17</td>
<td>Left Load Cell Step Response Data</td>
<td>85</td>
</tr>
<tr>
<td>4.18</td>
<td>Closeup of a Left Load Cell Step Response Occurring</td>
<td>86</td>
</tr>
<tr>
<td>4.19</td>
<td>Closeup of a Right Load Cell Step Response Occurring</td>
<td>87</td>
</tr>
<tr>
<td>4.20</td>
<td>Force Transfer Function Matched to Left Load Signal Path Step Response</td>
<td>89</td>
</tr>
<tr>
<td>4.21</td>
<td>Force Transfer Function Matched to Right Load Signal Path Step Response</td>
<td>90</td>
</tr>
<tr>
<td>4.22</td>
<td>Raw and Filtered Steady State Position Velocity and Acceleration</td>
<td>100</td>
</tr>
<tr>
<td>Figure</td>
<td>Description</td>
<td>Page</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
<td>------</td>
</tr>
<tr>
<td>4.23</td>
<td>Position Command, Raw Response, and Estimation of Ocean Waves</td>
<td>103</td>
</tr>
<tr>
<td>4.24</td>
<td>Closeup of Position Command, Raw Response, and Estimation of Ocean Waves</td>
<td>104</td>
</tr>
<tr>
<td>4.25</td>
<td>Velocity Raw Response and Estimation of Ocean Waves</td>
<td>105</td>
</tr>
<tr>
<td>4.26</td>
<td>Closeup of Velocity Raw Response and Estimation of Ocean Waves</td>
<td>106</td>
</tr>
<tr>
<td>4.27</td>
<td>Acceleration Raw Response and Estimation of Ocean Waves</td>
<td>107</td>
</tr>
<tr>
<td>4.28</td>
<td>Closeup of Acceleration Raw Response and Estimation of Ocean Waves</td>
<td>108</td>
</tr>
<tr>
<td>4.29</td>
<td>Force Raw Response and Estimation of Ocean Waves Note: ‘Raw’ Signal can not be seen because of Kalman filter almost perfectly matching raw signal.</td>
<td>109</td>
</tr>
<tr>
<td>4.30</td>
<td>Closeup of Force Raw Response and Estimation of Ocean Waves Note: ‘Raw’ Signal can not be seen because of Kalman filter almost perfectly matching raw signal.</td>
<td>110</td>
</tr>
</tbody>
</table>
## LIST OF TABLES

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>Motor Data Used In Calibration Check</td>
<td>26</td>
</tr>
<tr>
<td>3.2</td>
<td>Left Load Cell Tension Measurements</td>
<td>29</td>
</tr>
<tr>
<td>3.3</td>
<td>Right Load Cell Tension Measurements</td>
<td>30</td>
</tr>
<tr>
<td>3.4</td>
<td>Left Load Cell Compression Measurements</td>
<td>30</td>
</tr>
<tr>
<td>3.5</td>
<td>Right Load Cell Compression Measurements</td>
<td>31</td>
</tr>
<tr>
<td>4.1</td>
<td>Covariance Matrix - 1 second</td>
<td>52</td>
</tr>
<tr>
<td>4.2</td>
<td>Covariance Matrix - 10 second</td>
<td>53</td>
</tr>
<tr>
<td>4.3</td>
<td>Covariance Matrix - 60 second</td>
<td>53</td>
</tr>
<tr>
<td>4.4</td>
<td>Covariance Matrix - 200 second</td>
<td>54</td>
</tr>
<tr>
<td>4.5</td>
<td>Covariance Matrix - 1200 second</td>
<td>54</td>
</tr>
<tr>
<td>4.6</td>
<td>Covariance Comparison- 1 to 200 second</td>
<td>55</td>
</tr>
<tr>
<td>4.7</td>
<td>Covariance Comparison- 10 to 200 second</td>
<td>56</td>
</tr>
<tr>
<td>4.8</td>
<td>Covariance Comparison- 60 to 200 second</td>
<td>56</td>
</tr>
<tr>
<td>4.9</td>
<td>Covariance Comparison- 200 to 200 second</td>
<td>57</td>
</tr>
<tr>
<td>4.10</td>
<td>Covariance Comparison- 1200 to 200 second</td>
<td>57</td>
</tr>
<tr>
<td>4.11</td>
<td>Variance and Average of Signals in Static State</td>
<td>101</td>
</tr>
<tr>
<td>4.12</td>
<td>Static State Signal Improvements</td>
<td>101</td>
</tr>
</tbody>
</table>
## LIST OF APPENDICES

<table>
<thead>
<tr>
<th>Appendix</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>.1</td>
<td>Load Cell Test Sheet</td>
<td>119</td>
</tr>
<tr>
<td>.2</td>
<td>Computer Configurations</td>
<td>120</td>
</tr>
<tr>
<td>.2.1</td>
<td>Hardware Setup for Position Transfer Function Characterization</td>
<td>120</td>
</tr>
<tr>
<td>.2.2</td>
<td>Hardware Setup for Force Transfer Function Characterization</td>
<td>120</td>
</tr>
<tr>
<td>.2.3</td>
<td>Kalman Filter Tuning and Testing</td>
<td>121</td>
</tr>
<tr>
<td>.3</td>
<td>Position Transfer Function Characterization</td>
<td>123</td>
</tr>
<tr>
<td>.3.1</td>
<td>Simulink Model</td>
<td>123</td>
</tr>
<tr>
<td>.3.2</td>
<td>ControlDesk Interface</td>
<td>124</td>
</tr>
<tr>
<td>.3.3</td>
<td>MATLAB Code for Initialization of Simulink Blocks</td>
<td>124</td>
</tr>
<tr>
<td>.3.4</td>
<td>MATLAB Code for Calculation of Position Bode Plot</td>
<td>125</td>
</tr>
<tr>
<td>.4</td>
<td>Force Transfer Function Characterization</td>
<td>135</td>
</tr>
<tr>
<td>.4.1</td>
<td>Simulink Model</td>
<td>135</td>
</tr>
<tr>
<td>.4.2</td>
<td>ControlDesk Interface</td>
<td>136</td>
</tr>
<tr>
<td>.4.3</td>
<td>MATLAB Code for Initialization of Simulink Blocks</td>
<td>136</td>
</tr>
<tr>
<td>.4.4</td>
<td>MATLAB Code for Calculation of Force Transfer Function</td>
<td>137</td>
</tr>
<tr>
<td>.5</td>
<td>Noise Covariance Characterization</td>
<td>146</td>
</tr>
<tr>
<td>.5.1</td>
<td>Simulink Model</td>
<td>146</td>
</tr>
<tr>
<td>.5.2</td>
<td>ControlDesk Interface</td>
<td>147</td>
</tr>
<tr>
<td>.5.3</td>
<td>MATLAB Code for Calculation of Noise Covariance</td>
<td>147</td>
</tr>
<tr>
<td>.6</td>
<td>MATLAB Code for Calculation of Kalman Filter</td>
<td>151</td>
</tr>
<tr>
<td>.7</td>
<td>Kalman Filter Computer</td>
<td>159</td>
</tr>
<tr>
<td>.7.1</td>
<td>Simulink Model</td>
<td>159</td>
</tr>
<tr>
<td>.7.2</td>
<td>ControlDesk Interface</td>
<td>163</td>
</tr>
<tr>
<td>.7.3</td>
<td>MATLAB Code for Initialization of Simulink Blocks</td>
<td>163</td>
</tr>
<tr>
<td>.8</td>
<td>Sine and Ocean Drive Control Computer</td>
<td>166</td>
</tr>
<tr>
<td>.8.1</td>
<td>Simulink Model</td>
<td>166</td>
</tr>
<tr>
<td>.8.2</td>
<td>ControlDesk Interface</td>
<td>169</td>
</tr>
<tr>
<td>.8.3</td>
<td>MATLAB Code</td>
<td>170</td>
</tr>
<tr>
<td>Figure</td>
<td>Description</td>
<td>Page</td>
</tr>
<tr>
<td>--------</td>
<td>------------------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>1</td>
<td>Suggested 5000 lb Load Cell Test Sheet</td>
<td>119</td>
</tr>
<tr>
<td>2</td>
<td>Position Transfer Function Computer Configuration</td>
<td>120</td>
</tr>
<tr>
<td>3</td>
<td>Force Transfer Function Computer Configuration</td>
<td>121</td>
</tr>
<tr>
<td>4</td>
<td>Kalman Filter Tuning and Testing Computer Configuration</td>
<td>122</td>
</tr>
<tr>
<td>5</td>
<td>Position Transfer Function Simulink Model</td>
<td>123</td>
</tr>
<tr>
<td>6</td>
<td>Position Transfer Function ControlDesk Interface</td>
<td>124</td>
</tr>
<tr>
<td>7</td>
<td>Force Transfer Function Simulink Model</td>
<td>135</td>
</tr>
<tr>
<td>8</td>
<td>Force Transfer Function ControlDesk Interface</td>
<td>136</td>
</tr>
<tr>
<td>9</td>
<td>Noise Characterization Simulink Model</td>
<td>146</td>
</tr>
<tr>
<td>10</td>
<td>Noise Characterization ControlDesk Interface</td>
<td>147</td>
</tr>
<tr>
<td>11</td>
<td>Kalman Filter Computer Simulink Model - Top Level</td>
<td>159</td>
</tr>
<tr>
<td>12</td>
<td>Kalman Filter Computer Simulink Model - Signal Scaling Block for Pos. Vel. and Acc.</td>
<td>160</td>
</tr>
<tr>
<td>13</td>
<td>Kalman Filter Computer Simulink Model - Signal Scaling Block Load Cell Force</td>
<td>161</td>
</tr>
<tr>
<td>14</td>
<td>Kalman Filter Computer Simulink Model - Kalman Filter Block</td>
<td>162</td>
</tr>
<tr>
<td>15</td>
<td>Kalman Filter Computer ControlDesk Interface</td>
<td>163</td>
</tr>
<tr>
<td>16</td>
<td>Sine and Ocean Drive Computer Simulink Model - Top Level</td>
<td>166</td>
</tr>
<tr>
<td>17</td>
<td>Sine and Ocean Drive Computer Simulink Model - Signal Limits Check Block</td>
<td>167</td>
</tr>
<tr>
<td>18</td>
<td>Sine and Ocean Drive Computer Simulink Model - Limit Check Block</td>
<td>167</td>
</tr>
<tr>
<td>19</td>
<td>Sine and Ocean Drive Computer Simulink Model - Latch High Block</td>
<td>168</td>
</tr>
<tr>
<td>20</td>
<td>Sine and Ocean Drive Computer Simulink Model - Sine Wave Generator Block</td>
<td>168</td>
</tr>
<tr>
<td>Figure</td>
<td>Description</td>
<td>Page</td>
</tr>
<tr>
<td>--------</td>
<td>------------------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>21</td>
<td>Sine and Ocean Drive Computer Simulink Model - Ocean Wave Generator Block</td>
<td>169</td>
</tr>
<tr>
<td>22</td>
<td>Sine and Ocean Drive Computer ControlDesk Interface - Ocean Wave</td>
<td>169</td>
</tr>
<tr>
<td>23</td>
<td>Sine and Ocean Drive Computer ControlDesk Interface - Sine Wave</td>
<td>170</td>
</tr>
</tbody>
</table>
# LIST OF APPENDIX TABLES

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Cable Pin-Out for Control Computer to Load Cell Signal Amplifier</td>
<td>135</td>
</tr>
</tbody>
</table>
Chapter 1 – Introduction

The ocean is an untapped resource of renewable energy that holds the potential to help meet the world’s growing energy needs. Ocean waves, tidal water exchanges, and ocean currents are three areas which possess great hydro energy resources, and are currently receiving heavy focus from scientists, researchers, developers, and engineers.

Research at Oregon State University (OSU) has focused on point-absorber ocean wave energy generators as a solution to the question of how to generate energy from the ocean. Much of this research has focused on linear direct-drive generator technology, which potentially provides an elegant solution in its straight forward mechanical design and electrically efficient extraction of energy. The cost, remote nature of device placement, and harsh ocean environment place the demands of robustness and long-life on any ocean energy generating device. While fortitude to withstand the harsh ocean environment can only truly be assessed by actual ocean testing of a generating device, laboratory based testing of ocean energy generators is a critical intermediate step that yields ocean energy generator solutions rapidly and cost effectively.

To test point-absorber ocean wave energy generators at OSU, a machine has been designed and built called the Ocean Wave Energy Linear Test Bed. This Linear Test Bed has the ability to actuate linear ocean wave energy generators to simulate their vertical movement similar to that experienced from ocean waves. Since this CNC based automation was custom built for this ocean energy generator testing applica-
tion, it should be able to perform as such. This thesis focuses on the advancement of control and performance accuracy of the Linear Test Bed. Its ability to accurately replicate and simulate device testing as if it were in the ocean will aid in more rapid and accurate development of ocean wave energy generating devices.

1.1 Research Focus

The Linear Test Bed is an open-loop position controlled machine. For it to accurately simulate buoy movement in the ocean, it must be able to simulate the force interactions of the buoy with the ocean waves. To implement wave based force control on the Linear Test Bed, a closed-loop feedback control scheme is currently being developed. The Linear Test Bed is able to feedback the physical parameters a buoy generator actively experiences while being tested, specifically the buoy's position, velocity, acceleration, and experienced force in the Linear Test Bed machine.

1.1.1 Load Cell Calibration

To maintain machine simulation accuracy, the Linear Test Bed sensors must be calibrated to ensure accurate feedback signal measurement. Load cells, which measure the force a test buoy is experiencing, will be calibration checked. The calibration check results will be analyzed, and future calibration procedure suggestions are laid out.
1.1.2 Kalman Filtration of Feedback Signals

To implement force control of the Linear Test Bed, the position, velocity, acceleration, and force feedback signals must be used to create a closed-loop controller. Due to noise present in the system’s signals, it has not been possible to obtain accurate feedback signals. To solve this problem, a Linear Quadratic Gaussian Estimator, which is also known as a Kalman filter, will be used to construct zero phase delay filtered estimates of the Linear Test Bed’s feedback signals. The construction of the filter will depend on analysis of the feedback signals and characterization of the machine’s transfer functions.

1.2 MSRF/WESRF

The Ocean Wave Energy Linear Test Bed is currently located in the Wallace Energy Systems and Renewables Facility.

In 1996, the Motor Systems Research Facility (MSRF), headed by the late Dr. Allan Wallace, began research operations. In the summer of 2007, the MSRF was renamed the Wallace Energy Systems and Renewables Facility in honor of Dr. Wallace, and to help emphasize the renewable energy aspects of research conducted in the lab. WESRF is currently directed by Dr. Annette von Jouanne and Dr. Ted Brekken, who also manage the Energy Systems research group, a division of the School of Electrical Engineering and Computer Science[1].

WESRF has a variety of power and motor analysis equipment available for research. In addition to the Ocean Wave Energy Linear Test Bed, which is capable of
mechanically driving 20 kW into a device under test, WESRF also has a 750 kVA
dedicated power supply, 300hp rotary test bed, 120 kVA fully programmable source
with arbitrary waveform generator, a 50 kWh zinc bromine flow cell battery, and a
variety of precision measurement devices[2].

1.3 NNMREC

Development of the Ocean Wave Energy Linear Test Bed is currently funded through
the Department of Energy and the Northwest National Marine Renewable Energy
Center.

The Northwest National Marine Renewable Energy Center (NNMREC) is one
of two national marine renewable energy centers. Specifically NNMREC is a part-
nership formed between Oregon State University and University of Washington to
study ocean-based renewable energy, with funding coming from a variety of sources
including the United States Department of Energy.

Facilities for the Oregon based portion of NNMREC are at the Hatfield Ma-
rine Science Center, Wallace Energy Systems and Renewables Facility, and the O. H.
Hinsdale Wave Research Laboratory. These facilities and other associated researchers
focus on open ocean wave energy generation. Some examples of open ocean wave en-
ergy generation research include device characterization with the Wave Energy Linear
Test Bed, ocean wave test berth development, and ocean and lab based wave flume
testing of ocean wave energy generating devices. This collaborative research draws
from many areas of academia such as electrical engineering, mechanical engineering,
chemical engineering, civil engineering and from the sociological study of ocean energy effects on people.
Chapter 2 – Wave Energy Linear Test Bed
Figure 2.1: Wave Energy Linear Test Bed (LTB) Installed at WESRF with L-10 Ocean Wave Linear Generator (10 kW) Mounted in LTB
2.1 Background

An image of the OSU Ocean Wave Energy Linear Test Bed, or simply Linear Test Bed (LTB), can be seen as the light blue-gray machine in Figure 2.1. The LTB is a machine designed to test point-source ocean wave energy generators in a lab environment. Figure 2.1 shows the linear testbed loaded with an ocean wave energy generator device to be tested, specifically the L-10 Ocean Wave Energy Linear Generator.

2.1.1 LTB Origin

The LTB was originally designed for WESRF by Mundt & Associates, Inc., an automation design company located in Scottsdale, Arizona. The LTB was built in 2007, with completion of installation at WESRF in Spring 2008.

2.1.2 Description of Motion

Figure 2.1 shows the LTB with an ocean energy generating device loaded inside. The ocean energy generating device loaded inside is known as the device under test (DUT). The DUT is composed of two main components, the buoy and the spar.

The spar is the dark grey cylinder that runs the entire span of the machines height. It is bolted to the baseplate of the LTB and remains stationary during dynamic tests. The buoy has been disconnected from the DUT, but the buoy’s magnet section remains as the light yellow fiberglass cylinder in the center of the image. The LTB is able to move the magnet section up and down relative to the spar, creating a relative
linear motion used to produce electricity.

The magnet section is coupled to the LTB carriage yoke, and the LTB carriage yoke connects to the load cell suspension arms, and ultimately, the LTB carriage. The carriage yoke is the horizontally mounted white octagonal box-tube structure connected to the top of the central magnet section. The LTB carriage is connected through a series of belts and cogs to the LTB rotary drive motor, which drives the LTB carriage up and down linearly.
2.1.3 L-10 Generator

Figure 2.2: Solidworks Model of Complete L-10 Wave Energy Generator[3]
The L-10 generator is a point source ocean wave energy generator. It is designed to have 10 kW generating capacity. The generator section was originally designed by Joe Prudell, while Columbia Power Technologies Inc., and other graduate students contributed significantly to other design aspects such as the buoy and ballast tank. The L-10 generator experienced sea trials off of the coast of Newport, Oregon in Summer 2008.

The magnet section of the generator is denoted by the light blue cylinder which surround the central spar. The small gray cylinder in the center of the central spar represents the generator coil and back-iron. The large diameter yellow saucer (the buoy) and bottom ballast tank (connected to the spar) were disconnected for WESRF lab testing. The L-10 spar with coils and blue magnet section remain for testing in the LTB.

The L-10 generator was mounted in the LTB during system characterization and testing for Kalman filter construction.

The L-10 generator was not physically connected to the load cells during the load cell calibration check procedure.

2.1.4 Past Research

Past research relating to the LTB has been performed on two fronts. The first front is one of development for the machine itself. The bulk of this research was performed by two individuals, namely Nathan R. Henshaw, and Peter M. Hogan, both of whom did Masters of Science thesis work on the machine. Peter Hogan’s work on the
machine involved initial LTB design and control methodologies. Nathan Henshaw’s work focused more closely on closed-loop force control algorithms for simulation of actual ocean forces on the device under test.

The second area of research performed with the LTB has been to use it for testing and characterization of Ocean Wave Energy Generating devices. This has been done mostly by graduate students working in the WESRF, and has comprised ten different ocean wave energy linear devices as of June 2010, with an eleventh device scheduled for test in the summer of 2010.

2.2 External Hardware and Software

In addition to the LTB control box which houses its own control electronics, control computer, and Rexroth motor drive, there is hardware external to the LTB that is used to implement a closed-loop control scheme. This hardware includes a computer with rapid prototyping hardware and software installed in it. This rapid prototyping system is used for closed-loop feedback control of the LTB. The research process utilized two external computers with rapid prototyping equipment and were used for LTB system evaluation, tuning, and testing. All MATLAB code, Simulink models, ControlDesk interfaces, and computer connection diagrams for various tests and system characterizations are included in the Appendix.
2.2.1 MATLAB and Simulink

MATLAB and Simulink are the primary development environments used to create control schemes for the LTB. MATLAB is a computational software program and programming language specializing in matrix manipulation, but is capable of a wide variety of signal processing, communication, and other mathematically intensive applications. Simulink is a software package addition to MATLAB that allows block-diagram style construction of mathematical models. Realtime mathematical models can be constructed that interact with the physical domain through the combination of MATLAB, Simulink, a dSPACE rapid prototyping system (discussed in Section 2.2.2). The mathematical blocks which allow this are shown in Figure 2.4. Realtime processing of these mathematical models occurs in a discrete time frame. The calculation and I/O sample time used for most LTB analysis and control in this thesis was 10,000 samples per second.
2.2.2 Rapid Prototyping System - dSPACE DSP and ControlDesk

Each of the external control computers used to perform research (including closed-loop control of the LTB) have a rapid prototyping system installed, which includes a specialized digital signal processor (DSP) board made by the German company, dSPACE Incorporated. The dSPACE DSP used is model DS1103, which has 20x 16-bit analog-to-digital converters (ADC), 8x 16-bit digital-to-analog (DAC) converters, and 32 bits of digital I/O. The ADCs receive feedback signals from the LTB.
representing position, velocity, acceleration, left load cell force, and right load cell force. These feedback signals must maintain the ADC range limits of +/- 10 volts. A single DAC is used to send a commanded position voltage signal, which is also limited to +/- 10 volts. The amplitude of the commanded position voltage signal is directly proportional to LTB position, in that +/- 1 volt is equal to +/- 0.1 meters of vertical position movement from the homed center zero position of motion (1/2 point for range of LTB motion). The dSPACE digital I/O is used to send and receive system-enable safety commands between the external control computer and the LTB.

Figure 2.4: dSPACE DS1103 Rapid Prototyping System Internal Block Diagram[4]
Interaction with the dSPACE DSP is possible through a dSPACE program called ControlDesk. ControlDesk allows real-time access to the dSPACE DSP’s internal registers. ControlDesk’s ability to gain access to these registers allows for numerical manipulation of the DSP program. The DSP registers may also be read in real-time by ControlDesk, allowing display of real-time and real-world program numerical information in a user friendly interface. An example of the customizable ControlDesk user interface is shown in Figure 2.5.

Figure 2.5: ControlDesk Layout Example for Interface with dSPACE DSP[5]
Chapter 3 – Load Cell Calibration

The Load Cell Calibration chapter is comprised of three parts.

The first part discusses load cell functionality. Load cells have an internal resistive strain gauge sensor that measures the load cell’s applied force. Load cells measure the force that a test buoy is experiencing inside the LTB.

The second part is a ‘Partial Calibration Check’, as performed on the LTB. This partial check should not be considered an actual load cell calibration, in part because the partial check was comprised of an incomplete set of calibration check points. A complete calibration point set will include the capacity limits of the load cells, which is 5000 lbs for the load cells in question. It does, however, bring to light changes in the load cell force sensor measurement drift over time.

The last part is a recommended calibration procedure, which discusses the calibration methodology followed with the load cell creator - Interface Force Inc. The recommended calibration procedure also discusses different ways of implementing the procedure for the LTB in the WESRF.
3.1 Load Cell Description

![Image of Load Cell](image.png)

Figure 3.1: Interface Force 1000 Series Load Cell [6]

The LTB has two load cells it uses to measure the force the buoy is experiencing inside the machine as the buoy is moved up and down the spar. Figure 3.1 shows the load cell itself, while Figure 3.2 shows the load cell as an integral part of the load cell suspension arm. The load cell suspension arm is a means of coupling the motor driven LTB carriage to the buoy being tested. Being an integral part of this suspension arm allows the load cells to accurately measure forces between the LTB and the buoy. There are two of these load cell suspension arms installed on the LTB, that are used in parallel to support both the weight and movement of the buoy.
3.1.1 Internal Strain Gauge Sensor of Load Cell

The load cells installed on the LTB have an internal strain gauge which forms the actual physical sensor of the load cell itself. Interface Force load cells use a proprietary alloy for their load cell strain gauges. The alloy assists in temperature compensation to minimize temperature-dependent swing of the output voltage.

Typically, a strain gauge consists of a resistive sensor made from a metallic foil
pattern. The resistive sensor is then placed into a Wheatstone bridge, as seen in Figure 3.3. When the strain gauge is in a neutral state, as shown in Figure 3.4(a), the resistance of the strain gauge matches that of the other resistors in the Wheatstone Bridge, making the output voltage $V_O$ zero.

When the LTB accelerates a DUT downward, the strain gauge is compressed. This compression shortens the electron path through the resistive strain gauge, causing a lower resistance. Figure 3.4(b) shows an example of this compression.

The LTB accelerating a DUT upward causes strain gauge tension. This stretches the strain gauge, causing an increase in the electron path length, and an increased resistance. Figure 3.4(c) shows an example of this tension.

The output voltage relationship for the Wheatstone bridge can be seen in Equation 3.1, where $R_F$ represents fixed resistors, $R_S$ represents the strain gauge, $V_S$ is the DC
supply voltage, and $V_o$ represents the output voltage.

$$V_o = V_s \left( \frac{1}{2} - \frac{R_S}{R_S + R_F} \right)$$ (3.1)
(a) No Load State - Matched Resistance

(b) Compression State - Low Resistance

(c) Tension State - High Resistance

Figure 3.4: Strain Gauge Resistance States for Load Cell Wheatstone Bridge[8]
3.2 Partial Calibration Check

3.2.1 Hardware Configuration

For reasons of both safety and simplicity, the L-10 generator remained inside the LTB during calibration of the left and right load cells. The open cavity in the LTB to either side of the L-10 generator made independent testing of the load cells possible without removal of the L-10 wave generator.

The carriage yoke, which connects the LTB carriage to the buoy, is shown in Figure 2.2 as the dark blue rectangle on top of the buoy. This carriage yoke connects the LTB carriage to the buoy device under test. For the calibration test, the carriage yoke was disconnected from both the buoy and the magnet section, and set on the floor of the LTB, surrounding the L-10 generator spar. In disconnecting the yoke, the load cell suspension arms were uncoupled from the buoy device under test, freeing the arms for independent calibration testing.

To test a load cell in a state of tension, a downward force was applied to the load cell suspension arm yoke connection point. This yoke connection point is represented as the red gimbal mounting block in the lower right corner of Figure 3.2. The load cell arm with gimbal mounting point for tension testing is shown in Figure 3.8.
To test a load cell in a state of compression, an upward force was applied to the load cell suspension arm carriage connection point. The load cell arm with gimbal mounting point for compression testing is shown in Figure 3.6.
With the carriage yoke disconnected and the gimbal mounting point oriented for either tension or compression testing, calibration checking began. To perform the
tension test, motors of known weights were hung from the gimbal mount. The motors used for tension testing the left and right load cells are listed in Table 3.1. Each motor weight was measured with an S-type load cell as shown in Figure 3.7. An example image of a motor hanging from the gimbal mount on the load cell suspension arm is seen in Figure 3.8.

Table 3.1: Motor Data Used In Calibration Check

<table>
<thead>
<tr>
<th>Motor Used</th>
<th>Siemens Motor #</th>
<th>Name Plate</th>
<th>Measured</th>
</tr>
</thead>
<tbody>
<tr>
<td>50 HP</td>
<td>2023873001</td>
<td>700 lbs (3114 N)</td>
<td>732.5 lbs (3258 N)</td>
</tr>
<tr>
<td>100 HP</td>
<td>2023873002</td>
<td>1145 lbs (5093 N)</td>
<td>1130 lbs (5026 N)</td>
</tr>
<tr>
<td>200 HP</td>
<td>2023873003</td>
<td>1960 lbs (8719 N)</td>
<td>1963 lbs (8732 N)</td>
</tr>
</tbody>
</table>

To test the load cells in compression, the WESRF gantry crane was used as a fixed tension point. The load cell was coupled to the gantry crane, and then commanded downward away from the crane, putting tension on the LTB load cell suspension arm. An S-type tension load cell connected in-between the gantry crane and the LTB suspension arm is shown in Figure 3.7. This compression test configuration can be seen in Figure 3.8.
Figure 3.7: S-Type Load Cell and Associated Digital Readout Used in Compression Testing
The S-type load cell used in testing was made by VMC California Company, model number VLC-110, and serial number 592650. It is a 5000 lb capacity load cell with
operating point $3.0004 \frac{mV}{V}$.

3.2.2 Left and Right Load Cell Measurements Taken

All tension and compression tests were performed on 2009 December 17.

Tables 3.2, 3.3, Tables 3.4, and 3.5 list tension and compression measurements taken on left and right LTB load cells. Any measurements taken in pounds were converted to newtons with the conversion factor 4.4482216 Newtons to 1 lb.

Before tests, Left Load Cell (arm without block) zero offset: 11.5 Newtons. Arm with block 24.5 newtons before first motor loaded in.

Table 3.2: Left Load Cell Tension Measurements

<table>
<thead>
<tr>
<th>Motor Used</th>
<th>Expected Force</th>
<th>Force Measured</th>
<th>Load Zero Offset</th>
</tr>
</thead>
<tbody>
<tr>
<td>50 HP</td>
<td>3258 N</td>
<td>3067 N</td>
<td>—</td>
</tr>
<tr>
<td>100 HP</td>
<td>5026 N</td>
<td>5058 N</td>
<td>23.6 N</td>
</tr>
<tr>
<td>200 HP</td>
<td>8732 N</td>
<td>8748 N</td>
<td>24.4 N</td>
</tr>
</tbody>
</table>

Before tests, Right Load Cell (arm without block) zero offset: 13.8 Newtons. Arm with block 24.8 newtons before first motor loaded in.
Table 3.3: Right Load Cell Tension Measurements

<table>
<thead>
<tr>
<th>Motor Used</th>
<th>Expected Force</th>
<th>Force Measured</th>
<th>Post Zero Offset</th>
</tr>
</thead>
<tbody>
<tr>
<td>50 HP</td>
<td>3258 N</td>
<td>3064 N</td>
<td>24.7 N</td>
</tr>
<tr>
<td>100 HP</td>
<td>5026 N</td>
<td>5071 N</td>
<td>25.4 N</td>
</tr>
<tr>
<td>200 HP</td>
<td>8732 N</td>
<td>8730 N</td>
<td>24.8 N</td>
</tr>
</tbody>
</table>

Before test, Left Load Cell (arm with block), precompression measurement: 22.1 Newtons and After test, zero offset L-arm w/ no block 3.6 Newtons

Table 3.4: Left Load Cell Compression Measurements

<table>
<thead>
<tr>
<th>S-Type Load Cell Measurement</th>
<th>LTB Load Cell Measurement</th>
</tr>
</thead>
<tbody>
<tr>
<td>604 lbs (2687 N)</td>
<td>2670 N</td>
</tr>
<tr>
<td>1210 lbs (5382 N)</td>
<td>5353 N</td>
</tr>
<tr>
<td>1806 lbs (8033 N)</td>
<td>7996 N</td>
</tr>
<tr>
<td>2400 lbs (10676 N)</td>
<td>10610 N</td>
</tr>
</tbody>
</table>

Before test, Right Load Cell (arm with block), precompression measurement: 26.5 N. After test, zero offset L-arm with block 18.5 newtons
Table 3.5: Right Load Cell Compression Measurements

<table>
<thead>
<tr>
<th>S-Type Load Cell Measurement</th>
<th>LTB Load Cell Measurement</th>
</tr>
</thead>
<tbody>
<tr>
<td>611 lbs (2718 N)</td>
<td>2683 N</td>
</tr>
<tr>
<td>1207 lbs (5369 N)</td>
<td>5314 N</td>
</tr>
<tr>
<td>1800 lbs (8007 N)</td>
<td>7917 N</td>
</tr>
<tr>
<td>2410 lbs (10720 N)</td>
<td>10582 N</td>
</tr>
</tbody>
</table>

3.2.3 Error Analysis

To meet American Society of Test and Measurement (ASTM) International Standard E4-08, the load cell force measurements must lie within a 1% bound of 100% measurement accuracy[9]. The LTB does not need to meet standard ASTM E4-08 for WESRF operation, but ASTM E4-08 does provide a good general static calibration guideline for the LTB load cells. Also, this error analysis section does not constitute the error analysis of a complete calibration procedure, which would require more test points extending to the capacity limits of the load cells.
Figure 3.9: Partial Calibration Check Load Cell Percentage Accuracy

Figure 3.9 graphically shows the accuracy of the load cell measurements taken, and the 1% bounding lines as mentioned in ASTM E4-08 standard. The first three points on each plot indicate tension measurement accuracy, while the remaining four points on each plot indicate compression measurement accuracy.

For both the left and right load cells, the first two tension test points (-8748 N, -5058 N, -8730 N, -5071 N) are within desired the 1% bounding lines. The third test points (-3067 N and -3064 N) are outside of the bounding lines. This error on both of the third test points is symmetrically off by approximately the same percentage, and therefore the error will be attributed to operator error during the calibration check process.

The remaining four points for the left load cell are within the 1% bounding line,
while the right load cell compression measurements are all slightly outside the 1% bounding line. The right load cell compression inaccuracy is due to the way in which the right load cell was calibration checked.

To put the right load cell into compression, the gantry crane was used as a fixed point to pull down from. The gantry crane, however, was not physically able to directly align vertically above the top of the right load cell suspension arm. This caused side loading of the right load cell suspension arm, which slightly reduced the total vertical force load experienced by the load cell. In an attempt to alleviate this side loading, a strap and shackle were used to assist in vertical alignment of the gantry crane and the load cell suspension arm. This can be seen in Figure 3.10.
Figure 3.10: Partial Cause of Right Load Cell Compression Test Inaccuracy
Since the gantry crane can not physically be in the ideal spot to eliminate side loading of the load cell suspension arm, a slightly different method of elastic load cell calibration is suggested in Section 3.3.2.

### 3.2.4 Best Fit Corrections

Graphs were constructed of the expected force values versus actual force values. These expected versus measured values are represented as circled data points in Figures 3.11 and 3.12. These two figures also show the straight line approximation of these data points.

![Graph showing measured vs. actual force measurements for Left Load Cell](image)

Figure 3.11: Measured vs. Actual Force Measurements for Left Load Cell
The Least Squares method was used to fit a slope-intercept straight line approximation to the load cell data. The resulting straight line approximation formulas are given in Equations 3.2 and 3.3.

\[ Y_{LLC} = 1.005x - 5.758 \]  
\[ Y_{RLC} = 1.009x + 11.74 \]  

In Equations 3.2 and 3.3, the slope values (1.005 and 1.009) act as scalar multipliers for the incoming S.I. unitized force signals in the external computer. The constants being added (−5.758 and 11.74) serve as offset values to be added to the S.I. unitized
force signals in the external force computer. In other words, the incoming force signal (in S.I. units) can be fed into the $x$ variable of the equations to more accurately scale the force signal to its actual value.

The coefficient of determination, $R^2$, gives the proportion of variability in the load cell measured data set to the slope-intercept straight line approximation. The coefficient of determination calculation is given in Equation 3.6. It is calculated using the total sum of squares, $SS_{tot}$, which is variance, and the residual sum of squares, $SS_{err}$ which represents the errors between the load cell data and the approximated slope-intercept line.

$$SS_{tot} = \sum_i (y_i - \bar{y})^2$$  \hspace{1cm} (3.4)

$$SS_{err} = \sum_i (y_i - f_i)^2$$  \hspace{1cm} (3.5)

$$R^2 = 1 - \frac{SS_{err}}{SS_{tot}}$$ \hspace{1cm} (3.6)

The coefficient of determination for both the left and right load cells is 0.9999. This shows that the slope-intercept lines of Equations 3.2 and 3.2 are a good fit for the load cell measured test data.

### 3.3 Recommended Calibration Procedure

Ideally, calibrating the load cells would be done with test devices, such as the L-10 generator, removed. The safest, and most cost effective calibration method is elastic
calibration (Section 3.3.2). Calibration of the LTB’s left and right load cells should be performed independently in both directions of tension and compression to capacity rating. In the case of the load cells discussed in this thesis, 5000 lbs is the capacity rating. Figure 3.13 shows the configurations necessary for tension and compression of the LTB load cells using an elastic calibration device such as the S-type load cell.

Figure 3.13: Compression and Tension with an Elastic Calibration Device

In order to eliminate any LTB mechanical issues (drive shaft twist, carriage twist, linear rail binding, etcetera) the following considerations and adaptations must be made to the LTB.

- A single load cell suspension arm mounting point must be added in the center of the LTB carriage. This will allow for a balanced load on the LTB carriage
during testing.

- A fixed point must be established on the LTB baseplate directly below and vertical to the yoke attachment point of the load cell suspension arm. This fixed point will be used for elastic calibration of the LTB load cells in tension.

- A fixed point must be established directly above and vertical to the yoke attachment point of the load cell suspension arm. This fixed point will be used for elastic calibration of the LTB load cells in compression. The fixed point can be created with either use of the gantry crane, or with a steel cross bar bolted to the top of the LTB. A steel cross bar is the best choice because of the ability to repeatedly bolt it in the same location on top of the LTB each time load cell calibration is performed.

With the following hardware changes in place, calibration of the load cells can commence. Evenly distributed test points must be chosen throughout the full tension and compression range of the load cell’s capacity. Using the bidirectional 5000 lb load cell as an example, suitable test points are (in pounds): -5000, -4000, -3000, -2000, -1000, 0, 1000, 2000, 3000, 4000, and 5000. Smaller load cells should use the same distribution of force test points. Note that the weight of the S-type load cell and any associated cabling may need to be subtracted from the total force measured at 0 lbs force in the tension test. A load cell calibration test sheet is included in Appendix Section .1 which lists force test points for the 5000 lb load cells.

To apply the various test point forces, the LTB must be put into manual jog position control. The manual jog position step size should be 0.001 meters. By using
this small step size in manual position control mode, the incremental application of
force can be slowly applied to the LTB load cells by the LTB operator. As Figure 3.13
shows, the LTB carriage must be jogged downward to apply compression to the load
cell being calibrated. Figure 3.13 also shows that the LTB carriage must be jogged
upward to apply tension to the load cell being calibrated.

Once the load cell calibration data has been gathered, error analysis can take
place to determine if the load cells are within an appropriate accuracy range, such as
within 1% of 100% measurement accuracy.

After error analysis is completed, corrections may be made to offset the known
load cell measurement errors. The least squares method will provide a sufficient
straight line approximation. The straight line approximation is then used to correct
the load cells applied force calculation (made by the external control computer) that
will adjust the measured load cell force and report it as accurately as possible.

3.3.1 Known Weights

Known weights are a set of weights to be incrementally applied to the load cells. This
incremental application of weights will allow the force to take on multiple values. A
stackable weight set consisting of two 500 lb, one 1000 lb, and one 3000 lb weight will
allow testing between zero and full capacity of the LTB incrementally. These known
weights would then be applied in an incremental combined order such as that given
in Section 3.3.3.
3.3.2 Elastic Calibration Device

The S-type load cell such as in Figure 3.7 is considered a unidirectional elastic calibration device. When using an elastic calibration device such as an S-type load cell, a separate external force must be applied in either tension or compression with the elastic calibration device serving as an inner-connection between the externally applied force and the load cell to be tested. A unidirectional elastic calibration device is safer and more cost effective for LTB load cell characterization.

3.3.3 Manufacturer Calibration Methodology

The American Society for Testing and Materials (ASTM) International standard E 4-08 discusses ‘Standard Practices for Force Verification of Testing Machines’. This is one of the standards of calibration followed by Interface Force Incorporated, making their calibration process National Institute of Standards and Technology (NIST) traceable. Interface Force Inc. uses an automated hydraulic load cell testing machine to perform the following calibration operations.

The calibration procedure starts when the load cell is measured in the zero (or neutral) force position. The load cell is actuated through five separate test points in between zero and full capacity. In the case of the LTB load cells, these test points may occur at 1000, 2000, 3000, 4000, and 5000 lbs. After measuring the 5000 lb capacity force, the load cell is returned to between 20% and 30% of its capacity load (1000 lbs for example). Then the load cell is returned to the neutral (zero) force point. The procedure is repeated in the opposite direction (tension versus compression), stepping
through five test points. Upon reaching load cell capacity in the opposite direction, the load cell is returned to between 20% and 30% of capacity and measured, and then finally back to neutral for the third and final zero force measurement. Each step of the calibration procedure allows for a 10 second settling time before the load cell is measured to allow for transient movement of the load cell to settle.

3.3.4 Dynamic Load Cell Testing

Dynamic testing involves determining load cell magnitude and phase response, in essence, the load cell transfer function (or Bode plot).

ASTM International standard E467-08 ‘Standard Practice for Verification of Constant Amplitude Dynamic Forces in an Axial Fatigue Testing System’ is the closest standard available from ASTM for frequency dependent magnitude response analysis. This analysis, however, does not account for characterization of phase delay of the load cell itself.

To test this, a nested sweep procedure would be followed which sweeps through the range of expected operating frequencies and swept through a range of load cell forces (up to load cell capacity). Currently, WESRF and Interface Force Inc. do not possess a means of dynamically testing the LTB load cells. Static testing of the load cells should suffice because of the simplifying assumptions made that the load cell sensor itself is a resistive device which is not frequency dependent in nature (as discussed in Section 4.4.4.1).
3.3.5 Calibration Location

Often load cells are sent to calibration test facilities for calibration testing. This is not the best option for the LTB, since it does not use load cells in the traditional sense. Often, load cells are used to take static force measurements by an operator observing a calibrated digital readout.

This calibrated digital readout is not present, but instead consists of the load cell signal path and force control computer. Specifically, a ‘Differential to Single Ended Signal Amplifier’, a ‘Low-Pass Filter Amplifier’, and the dSPACE Analog-to-Digital converter as can be seen in Figure 4.15.

If the performance characteristics of any one of these subsystems change over time, then drift in the force feedback signal could occur. The potential drift in signal accuracy from LTB subsystems gives strong consideration for calibrating the load cells when they are connected into the LTB system.
Chapter 4 – The Kalman Filter

This chapter talks specifically about how to create a Kalman filter for the LTB. This Kalman filter is needed because of excessively noisy feedback signals which have prevented accurate closed-loop force control of the LTB. The two types of noise introduced which are process noise (w) and measurement noise (v) are introduced into the system as seen in Figure 4.1.

![Figure 4.1: Process and Measurement Noise Introduced into System](image)

The resulting normalized feedback signals y of the plant can be seen in Figure 4.2. These noisy feedback signals provide a visual example of noise introduced into the plant which has limited the ability to create a functioning closed loop force control for the LTB. The red acceleration feedback signal has the visibly highest noise power.
4.1 Linear Quadratic Gaussian Controller

The Linear Quadratic Gaussian (LQG) control problem is a fundamental control problem for which the solution, the Linear Quadratic Gaussian Controller (LQGC) allows accurate control of a plant which contains a Gaussian (or normal) distribution of noise. The Linear Quadratic Gaussian Controller as shown in Figure 4.3 has three parts, the estimator (also known as the Kalman filter), the integrator, and the optimal...
gain (known as K).

![Figure 4.3: Linear Quadratic Gaussian Controller and Plant](image)

The ‘L.Q. Estimator’ (or Kalman Filter) block of Figure 4.3 accepts inputs $u$ (the commanded plant input) and $y$ (the plant sensor outputs). These signals are used by the Kalman filter to create estimates of the plant’s current states, $\hat{x}$, and to create estimates of the plant’s outputs, $\hat{y}$. The Kalman filter, if constructed properly, will remove much of the Gaussian distribution of ambient noise present on the plant sensor output estimate, $\hat{y}$.

The ‘Integrator’ block of Figure 4.3 accepts the input $r - y$ which represents the error difference between the commanded system input $r$ and the plant sensor outputs $y$. The integrator sends out this accumulated error rate via $x_i$ which is the total integrated error quantity up to a given point in time.

The L.Q. Optimal Gain (also known as ‘K’) performs the final control signal gain calculation to send to the system plant. The Optimal Gain block accepts the state estimates, $\hat{x}$, from the L.Q. Estimator block, and the integrated error, $x_i$, from the Integrator block. The plant command is then sent out of the L.Q. Controller as $u$ to the plant, and fed back to the L.Q. Estimator.
This thesis delves specifically into construction of the Linear Quadratic Gaussian Estimator, or Kalman filter, and its associated building blocks. LTB signal flow associated with the Kalman filter can be seen in Figure 4.4. The variable $y$ represents raw unfiltered signals from the LTB’s Rexroth motor drive and load cells. The variable $u$ represents the commanded input from the control computer, which, for example, could be a recorded wave profile. The variable $\hat{y}$ represents the estimated output signals from the LTB, while $\hat{x}$ represents the current state estimates of the LTB.

![Figure 4.4: L.Q.G. Estimator (Kalman Filter) and LTB Signals to be Filtered](image)

4.2 Background of the Kalman Filter

The mathematical method known as the Kalman filter is named after Rudolf E. Kalman, a Hungarian-American electrical engineer who co-invented the Kalman filter with R.S. Bucy in the 1960's[10]. Today, the Kalman filter is used in a wide variety of control applications such as rocket[11] and satellite control([12], [13], [14]), ocean-based ship positioning control ([15], [16], [17], [18]), aerial flight control ([19], [20]), and even for simple applications such as the phase-locked loop[21].
The Kalman Filter (or Kalman Estimator) provides a means of removing noise from the feedback signals from a system plant. In this way, the Kalman Filter provides a more accurate estimate of the actual system states and outputs than the system itself is able to provide via sensor measurements. This is due to removal of the Gaussian distribution of noise present on the plant outputs.

For the Kalman filter to accurately estimate the states of the plant, it needs, at a minimum, the signals from which output and state estimation are based: the plant input and output signals $u$ and $y$ (as mentioned in the LQGC section and shown in Figure 4.3.)

Additionally, to construct a Kalman filter, the following plant information is needed:

- The covariance matrix for the process noise $Q_N$, as measured from the plant.
- The covariance of the observation noise $R_N$, selected by the control designer.
- The plant’s physical response equations in state space format.

$R_N$, $Q_N$, and the plant’s physical response equations will be explained in the following sections.

4.3 Process Noise

The process noise measurement of the Kalman filter is essentially background noise the system in a static state is experiencing. It can also be thought of as ambient noise,
or white noise, and is unfortunately added into the feedback signals. This undesired process noise appears mixed in with the feedback signal \( \hat{y} \) in Figure 4.3.

Process noise is to be quantified on the \( \hat{y} \) feedback signal lines. To quantify this noise, the \( \hat{y} \) signal must not be changing due to system changes of state. For an electrical system, this would mean that the system has been allowed to reach DC steady state (except the ambient process noise).

Once the system has reached static equilibrium (no changes of state), measurements of the ambient noise must be taken for quantification purposes. Sampling rates and sampling resolutions should match those used by the system to read its feedback sensors.

With process noise measurements of each individual feedback signal taken, the covariance of the process noises must be calculated.

\[
\text{Covariance}(x, y) = \frac{1}{N} \sum_{i=1}^{N} (x_i - \bar{x})(y_i - \bar{y})
\]  

(4.1)

Covariance is a measurement of how much signals change relative to each other, and must be calculated for every combination of feedback signals possible. In Equation 4.1, \( x \) and \( y \) are separate data sets of ambient noise from two different feedback sensors.

If only one feedback sensor signal is being used, then \( x \) and \( y \) would represent the same data set, which will give the variance. Variance is a special case of covariance, in which the \( x \) and \( y \) values are the same (from the same data set). This singular feedback sensor covariance value will then be equal to \( R_N \), the process noise value.
which goes into compilation of the Kalman filter.

For a multiple sensor system, a square matrix must be formed, in which the row and column indices are representative of the sensor feedback signals coming from the plant. The diagonal values of the matrix will then be the variance values of the various feedback signals ($x$ and $y$ will be equal to each other.) This final matrix will then be equal to $R_N$, the process noise which goes into compilation of the Kalman filter.

4.3.1 Process Noise Measurement for the Linear Test Bed

Process noise must be measured for five separate signal lines on the LTB. The signal lines are position, velocity, acceleration, left load cell force, and right load cell force. These noise measurements must be taken without dynamic system response occurring that would reflect in dynamic (changing) feedback signals, skewing the process noise measurements of these feedback signal lines.

4.3.1.1 Noise Measurement Procedure

Process noise for the LTB was measured on the 5 separate feedback signal lines with a dSPACE DS1103 I/O Board. The left and right load cell signals were summed together to create a single force signal for the final process noise analysis. The sampling rates used were 1000 samples per second and 10,000 samples per second. The sample duration time varied between one second and 1200 seconds. The one second 10,000
sample-per-second noise measurement data for the LTB can be seen in Figure 4.5.

Figure 4.5: LTB Noise Measurements, 10,000 Samples, 1 Second Duration

To take the measurements with the external computer, the LTB was energized, homed, and prepared to receive operating commands. It was not put into ‘Force Control Mode’, to disable any chance of commanded change of state from non-moving steady state. Upon achieving this state with the position, velocity, acceleration, and force signals connected to the dSPACE computer, ControlDesk was then used to record the ambient noise on the feedback sensor lines.

4.3.1.2 Covariance Results From Noise Data

Five different combinations of sample times per durations were used to gather ambient noise data. The sampling durations and rates are as follows:
• 1 second sample duration, 10,000 samples per second.

• 10 second sample duration, 10,000 samples per second.

• 60 second sample duration, 10,000 samples per second.

• 200 second sample duration, 10,000 samples per second.

• 1200 second sample duration, 1,000 samples per second.

The largest data set taken was the 200 second sample duration, with 2,000,000 data points taken for each of the five feedback signals. This data set size was at the approximate limit of ControlDesk’s ability to record data. The 200 second sample data set is shown in the following set of tables, specifically Table 4.4.

Table 4.1: Covariance Matrix for Position, Velocity, Acceleration, and Force Measurements Taken: 1 second duration, 10000 samples per second
   From File: ‘static_noise_pvalr_10000persec_1sec.mat’

<table>
<thead>
<tr>
<th>$R_N$</th>
<th>Position</th>
<th>Velocity</th>
<th>Acceleration</th>
<th>Force</th>
</tr>
</thead>
<tbody>
<tr>
<td>Position</td>
<td>4.1198e-09</td>
<td>3.8910e-09</td>
<td>-6.0868e-07</td>
<td>4.1929e-09</td>
</tr>
<tr>
<td>Velocity</td>
<td>3.8910e-09</td>
<td>2.51079e-08</td>
<td>1.4334e-07</td>
<td>9.7915e-09</td>
</tr>
<tr>
<td>Acceleration</td>
<td>-6.0868e-07</td>
<td>1.4334e-07</td>
<td>3.3364e-03</td>
<td>-1.1636e-05</td>
</tr>
<tr>
<td>Force</td>
<td>4.1930e-09</td>
<td>9.7915e-09</td>
<td>-1.1636e-05</td>
<td>1.1999e-06</td>
</tr>
</tbody>
</table>
Table 4.2: Covariance Matrix for Position, Velocity, Acceleration, and Force Measurements Taken: 10 second duration, 10000 samples per second
From File: ‘static_noise_pvalr_10000persec_10sec.mat’

<table>
<thead>
<tr>
<th></th>
<th>Position</th>
<th>Velocity</th>
<th>Acceleration</th>
<th>Force</th>
</tr>
</thead>
<tbody>
<tr>
<td>R_N</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Position</td>
<td>4.0830e-09</td>
<td>4.1128e-09</td>
<td>-4.4817e-07</td>
<td>4.8854e-09</td>
</tr>
<tr>
<td>Velocity</td>
<td>4.1128e-09</td>
<td>2.6584e-08</td>
<td>3.6815e-08</td>
<td>1.0821e-08</td>
</tr>
<tr>
<td>Acceleration</td>
<td>-4.4817e-07</td>
<td>3.6815e-08</td>
<td>3.3958e-03</td>
<td>-1.5200e-05</td>
</tr>
<tr>
<td>Force</td>
<td>4.8854e-09</td>
<td>1.0821e-08</td>
<td>-1.5200e-05</td>
<td>1.3450e-06</td>
</tr>
</tbody>
</table>

Table 4.3: Covariance Matrix for Position, Velocity, Acceleration, and Force Measurements Taken: 60 second duration, 10000 samples per second
From File: ‘static_noise_pvalr_10000persec_60sec.mat’

<table>
<thead>
<tr>
<th></th>
<th>Position</th>
<th>Velocity</th>
<th>Acceleration</th>
<th>Force</th>
</tr>
</thead>
<tbody>
<tr>
<td>R_N</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Position</td>
<td>4.1594e-09</td>
<td>4.0107e-09</td>
<td>-5.1194e-07</td>
<td>4.4757e-09</td>
</tr>
<tr>
<td>Velocity</td>
<td>4.0107e-09</td>
<td>2.6224e-08</td>
<td>5.7757e-08</td>
<td>1.0959e-08</td>
</tr>
<tr>
<td>Acceleration</td>
<td>-5.1195e-07</td>
<td>5.7757e-08</td>
<td>3.4958e-03</td>
<td>-1.3852e-05</td>
</tr>
<tr>
<td>Force</td>
<td>4.4757e-09</td>
<td>1.0959e-08</td>
<td>-1.3852e-05</td>
<td>1.2916e-06</td>
</tr>
</tbody>
</table>
Table 4.4: Covariance Matrix for Position, Velocity, Acceleration, and Force Measurements Taken: 200 second duration, 10000 samples per second
From File: 'static_noise_pvalr_10000persec_200sec.mat'

<table>
<thead>
<tr>
<th>( R_N )</th>
<th>Position</th>
<th>Velocity</th>
<th>Acceleration</th>
<th>Force</th>
</tr>
</thead>
<tbody>
<tr>
<td>Position</td>
<td>3.7678e-09</td>
<td>3.1014e-09</td>
<td>-5.0424e-07</td>
<td>3.4143e-09</td>
</tr>
<tr>
<td>Velocity</td>
<td>3.1014e-09</td>
<td>2.3283e-08</td>
<td>2.4744e-08</td>
<td>8.3411e-09</td>
</tr>
<tr>
<td>Acceleration</td>
<td>-5.0424e-07</td>
<td>2.4744e-08</td>
<td>3.5126e-03</td>
<td>-1.2673e-05</td>
</tr>
<tr>
<td>Force</td>
<td>3.4143e-09</td>
<td>8.3411e-09</td>
<td>-1.2673e-05</td>
<td>1.2791e-06</td>
</tr>
</tbody>
</table>

Table 4.5: Covariance Matrix for Position, Velocity, Acceleration, and Force Measurements Taken: 1200 second duration, 1000 samples per second
From File: ‘static_noise_pvalr_1000persec_1200sec.mat’

<table>
<thead>
<tr>
<th>( R_N )</th>
<th>Position</th>
<th>Velocity</th>
<th>Acceleration</th>
<th>Force</th>
</tr>
</thead>
<tbody>
<tr>
<td>Position</td>
<td>3.8711e-09</td>
<td>3.2113e-09</td>
<td>-4.9234e-07</td>
<td>5.9744e-09</td>
</tr>
<tr>
<td>Velocity</td>
<td>3.2113e-09</td>
<td>2.3665e-08</td>
<td>2.9571e-08</td>
<td>1.1131e-08</td>
</tr>
<tr>
<td>Acceleration</td>
<td>-4.9234e-07</td>
<td>2.9571e-08</td>
<td>3.4994e-03</td>
<td>-1.1895e-05</td>
</tr>
<tr>
<td>Force</td>
<td>5.9744e-09</td>
<td>1.1131e-08</td>
<td>-1.1895e-05</td>
<td>1.4984e-06</td>
</tr>
</tbody>
</table>

Any one of the five preceding tables could be used for the \( R_N \) needed for construction of the Kalman filter.

4.3.1.3 Comparison of Covariance Matrices

The covariance matrix created from the '200 second' data set will be used as a baseline to compare the other four data sets. The '200 second' covariance matrix was chosen
as the baseline because it was constructed from the largest number of noise measurements. Computing the ratios of the other data sets to the baseline will produce ratios which dictate the differences in covariance.

In looking at the Comparison of Covariance Matrices to the base Covariance Matrix (‘200 second’ data set), the largest difference between covariance results is in the ‘1 second’ covariance matrix. The value which stands out the most is the Velocity to Acceleration // Acceleration to Velocity covariance, which is approximately 579% greater in the ‘1 second’ matrix. Comparing the other covariance values in the ‘1 second’ matrix to the ‘200 second’ matrix reveals that covariance values did not deviate more than approximately 20%.

Table 4.6: Comparison of Covariance Matrices
(Covariance from 1 second data) ÷ (Covariance from 200 second data)

<table>
<thead>
<tr>
<th>$\mathbf{R}_N \div \mathbf{R}_N$</th>
<th>Pos. ÷ Pos.</th>
<th>Vel. ÷ Vel.</th>
<th>Acc. ÷ Acc.</th>
<th>Force ÷ Force</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pos. ÷ Pos.</td>
<td>1.093</td>
<td>1.255</td>
<td>1.207</td>
<td>1.228</td>
</tr>
<tr>
<td>Vel. ÷ Vel.</td>
<td>1.254</td>
<td>1.078</td>
<td>5.793</td>
<td>1.174</td>
</tr>
<tr>
<td>Acc. ÷ Acc.</td>
<td>1.207</td>
<td>5.793</td>
<td>0.950</td>
<td>0.918</td>
</tr>
<tr>
<td>Force ÷ Force</td>
<td>1.228</td>
<td>1.174</td>
<td>0.918</td>
<td>0.938</td>
</tr>
</tbody>
</table>
Table 4.7: Comparison of Covariance Matrices
(Covariance from 10 second data) ÷ (Covariance from 200 second data)

<table>
<thead>
<tr>
<th>$\mathbf{R}_N \div \mathbf{R}_N$</th>
<th>Pos. ÷ Pos.</th>
<th>Vel. ÷ Vel.</th>
<th>Acc. ÷ Acc.</th>
<th>Force ÷ Force</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pos. ÷ Pos.</td>
<td>1.084</td>
<td>1.326</td>
<td>0.889</td>
<td>1.431</td>
</tr>
<tr>
<td>Vel. ÷ Vel.</td>
<td>1.326</td>
<td>1.142</td>
<td>1.488</td>
<td>1.297</td>
</tr>
<tr>
<td>Acc. ÷ Acc.</td>
<td>0.889</td>
<td>1.488</td>
<td>0.967</td>
<td>1.199</td>
</tr>
<tr>
<td>Force ÷ Force</td>
<td>1.431</td>
<td>1.297</td>
<td>1.199</td>
<td>1.052</td>
</tr>
</tbody>
</table>

The next lowest level of mismatch is between the ‘60 second’ data and the base data (‘200 second’). There is a 233% difference in the Velocity to Acceleration // Acceleration to Velocity covariance. Other covariance values for these data sets are reasonably well matched.

Table 4.8: Comparison of Covariance Matrices
(Covariance from 60 second data) ÷ (Covariance from 200 second data)

<table>
<thead>
<tr>
<th>$\mathbf{R}_N \div \mathbf{R}_N$</th>
<th>Pos. ÷ Pos.</th>
<th>Vel. ÷ Vel.</th>
<th>Acc. ÷ Acc.</th>
<th>Force ÷ Force</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pos. ÷ Pos.</td>
<td>1.104</td>
<td>1.293</td>
<td>1.015</td>
<td>1.311</td>
</tr>
<tr>
<td>Vel. ÷ Vel.</td>
<td>1.293</td>
<td>1.126</td>
<td>2.334</td>
<td>1.314</td>
</tr>
<tr>
<td>Acc. ÷ Acc.</td>
<td>1.015</td>
<td>2.334</td>
<td>0.995</td>
<td>1.093</td>
</tr>
<tr>
<td>Force ÷ Force</td>
<td>1.311</td>
<td>1.314</td>
<td>1.093</td>
<td>1.010</td>
</tr>
</tbody>
</table>
Table 4.9: Comparison of Covariance Matrices
(Covariance from 200 second data)\(\div\) (Covariance from 200 second data)

<table>
<thead>
<tr>
<th>(\mathbf{R}_N \div \mathbf{R}_N)</th>
<th>Pos.(\div)Pos.</th>
<th>Vel.(\div)Vel.</th>
<th>Acc.(\div)Acc.</th>
<th>Force(\div)Force</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pos.(\div)Pos.</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Vel.(\div)Vel.</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Acc.(\div)Acc.</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Force(\div)Force</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 4.10: Comparison of Covariance Matrices
(Covariance from 1200 second data)\(\div\) (Covariance from 200 second data)

<table>
<thead>
<tr>
<th>(\mathbf{R}_N \div \mathbf{R}_N)</th>
<th>Pos.(\div)Pos.</th>
<th>Vel.(\div)Vel.</th>
<th>Acc.(\div)Acc.</th>
<th>Force(\div)Force</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pos.(\div)Pos.</td>
<td>1.027</td>
<td>1.035</td>
<td>0.976</td>
<td>1.750</td>
</tr>
<tr>
<td>Vel.(\div)Vel.</td>
<td>1.035</td>
<td>1.016</td>
<td>1.195</td>
<td>1.334</td>
</tr>
<tr>
<td>Acc.(\div)Acc.</td>
<td>0.976</td>
<td>1.195</td>
<td>0.996</td>
<td>0.939</td>
</tr>
<tr>
<td>Force(\div)Force</td>
<td>1.750</td>
<td>1.334</td>
<td>0.939</td>
<td>1.171</td>
</tr>
</tbody>
</table>

The conclusion from comparing covariance values to the base data set is that in general, it should not matter significantly which of these data sets are used to create the \(\mathbf{R}_N\) matrix for the Kalman filter because the covariance values of the respective cells are all within the same order of magnitude of each other. There appears to be some slight imbalance in the Velocity to Acceleration // Acceleration to Velocity covariance calculation when comparing different covariance matrices, and based upon this, the matrix which has been created from the most data points, the ‘200 second’ matrix, should be used for Kalman analysis.
4.4 Transfer Function Characterization of the Linear Test Bed

There are five signals being fed back to the external control computer for which transfer function equations are needed. Two of these, the left and right load cells, are summed together to create a single force signal, which in turn, will reduce the number of actual signals being used for closed-loop feedback control to four.

These four signals must be represented in transfer function format, and ultimately in state space format for creation of the Kalman filter. The Kalman filter signal inputs and outputs can be seen in Figure 4.4.

Of the four transfer functions needed, velocity and acceleration can be easily determined by direct physical relationship. Velocity and acceleration are 1st and 2nd order time derivatives of the position. The other two transfer functions, position and force, must be derived empirically through direct testing of the LTB machine.

4.4.1 Velocity Transfer Function

Based upon the Newtonian physical relationship between an object’s velocity and position, the time derivative of position will always be equal to an object’s velocity (at least on this planet, and most of the others too). This idealized relationship is represented in the frequency domain as shown in Equation 4.2. In this equation, $V_M$ is the measured velocity of a buoy, and $Z_M$ is the measured position of the buoy.

$$V_M = sZ_M$$

(4.2)
While this equation holds true for the object’s physical movement, receiving this information electronically through feedback signals is inaccurate because of delays in the transmission system, inaccuracies in the way position and velocity are measured, inaccuracies in the time derivative calculations on fixed point (optical encoder) position data, sensor inaccuracies, etc.

To create a more accurate representation of the true position to velocity relationship, a transfer function $H_V$, was used to assist in modeling system disturbances. This modified equation, with the included transfer function, is shown in Equation 4.3. Its derivation toward state space format can be seen in Equations 4.4. In these equations, $V_{AM}$ is the approximate measured velocity, $W_V$ is the velocity 1st order transfer function pole frequency, while $V_M$ and $Z_M$ are the measured velocity and position from the LTB, respectively.

$$V_{AM} = sH_V Z_M$$
$$= s \frac{W_V}{s + W_V} Z_M$$
$$= \frac{W_V}{1 + \frac{W_V}{s}} Z_M$$

$$V_{AM} \frac{1}{1 + \frac{W_V}{s}} = W_V Z_M$$

$$V_{AM} + V_{AM} \frac{W_V}{s} = W_V Z_M$$

$$V_{AM} = W_V Z_M - \frac{V_{AM} W_V}{s} \quad (4.4)$$

To convert the velocity transfer function into a final format, a derivative/integral
trick is used to adapt the transfer function into a state space relatable format [22].

\[ d\left(\frac{V_{AM}}{s}\right) = -\frac{W_V V_{AM}}{s} + W_V Z_M \]  \hspace{1cm} (4.5)

Note that the desired output is \( V_{AM} \) which Equation 4.5 gives by taking the derivative of the integral, \( d\left(\frac{V_{AM}}{s}\right) \). This means that the state matrix \( A \) and output matrix \( C \) are equivalent, in addition to the input matrices \( B \) and feedforward \( D \) being exactly the same. The final state space representation of the velocity transfer function is given in Equation 4.6. The pole frequency \( W_V \) was determined through a tuning process discussed in Section 4.5.2. The final pole frequency in radians is \( 10\pi \).

\[
\begin{align*}
A &= \begin{bmatrix} -W_V \end{bmatrix}, & B &= \begin{bmatrix} W_V \end{bmatrix} \\
C &= \begin{bmatrix} -W_V \end{bmatrix}, & D &= \begin{bmatrix} W_V \end{bmatrix} \\
x &= \frac{V_{AM}}{s}, & y &= V_{AM}, & u &= Z_M
\end{align*}
\]  \hspace{1cm} (4.6)

4.4.1.1 Velocity Transfer Function - Alternate Derivation

An alternative method of creating the velocity transfer function (from position input) is as follows. Equation 4.7 shows the 1st order transfer function relationship \( H_C \) of
measured position $Z_M$ to commanded position $Z_C$ in the frequency domain.

\[ Z_M = Z_C H_C \]
\[ = Z_C \frac{W_C}{s + W_C} \]
\[ Z_M(s + W_C) = Z_C W_C \]
\[ sZ_M = Z_C W_C - Z_M W_C \quad (4.7) \]

Equation 4.8 gives the velocity $V_M$ as the time derivative of measured position $Z_M$ with associated velocity transfer function $H_V$.

\[ V_M = sZ_M H_V \]
\[ = s \frac{Z_M}{\frac{s}{W_V} + 1} \quad (4.8) \]

Equations 4.7 and 4.8 are then tied together in the final Equation 4.9, which easily transitions into state space format as shown in Equation 4.10.

\[ sV_M = sZ_M W_V - V_M W_V \]
\[ = W_P W_V Z_C - W_P W_V Z_M - V_M W_V \quad (4.9) \]
\[ A = \begin{bmatrix} -W_v \\ 0 \end{bmatrix}, \quad B = \begin{bmatrix} -W_pW_v & W_pW_v \end{bmatrix}, \quad C = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad D = \begin{bmatrix} 0 & 0 \end{bmatrix} \] (4.10)

\[ x = V_M, \quad y = V_M, \quad u = \begin{bmatrix} Z_M \\ Z_C \end{bmatrix} \]

### 4.4.2 Acceleration Transfer Function

As was pointed out in Section 4.4.1 on the derivation of the velocity transfer function, the acceleration transfer function has a similar derivation method. The basic Newtonian relationship between acceleration and velocity is shown in Equation 4.11. In this equation, \( A_M \) is the measured acceleration of a buoy, and \( V_M \) is the measured velocity of the buoy.

\[ A_M = sV_M \] (4.11)

As was the case with the calculation of velocity by the LTB, the calculation of acceleration is particularly fraught with error, especially the second order time derivative performed on fixed point position data to calculate the acceleration. Additional errors arise due to sensor measurement error, transmission losses, and other things that plague the delivery of the real-time acceleration value to the external control computer.

To account for these inaccuracies, a transfer function is included with Equation 4.11 as reflected in Equation 4.12. Its derivation toward state space format can
be seen in Equations 4.13. In these equations, \( A_{AM} \) is the approximate measured acceleration, \( W_A \) is the acceleration 1st order transfer function pole frequency, while \( A_M \) and \( V_M \) are the measured acceleration and velocity from the LTB, respectively.

\[
A_{AM} = sH_A V_M
\]

\[
= s \frac{W_A}{s + W_A} V_M
\]

\[
= \frac{W_A}{1 + \frac{W_A}{s}} V_M
\]

\[
A_{AM} \frac{1}{1 + \frac{W_A}{s}} = W_A V_M
\]

\[
A_{AM} + V_{AM} \frac{W_A}{s} = W_A A_M
\]

\[
A_{AM} = W_V Z_M - \frac{A_{AM} W_A}{s}
\] (4.13)

To convert the acceleration transfer function into a final format, a derivative/integral method is used to adapt the transfer function into a state space relatable format [22].

\[
d\left(\frac{A_{AM}}{s}\right) = -\frac{W_A A_{AM}}{s} + W_A V_M
\] (4.14)

Based upon the derivative/integral method for constructing the state space representation, the desired output is \( A_{AM} \) which Equation 4.14 gives by taking the derivative of the integral, \( d\left(\frac{A_{AM}}{s}\right) \). Again, the state matrix (A) and output matrix (C) are equivalent. Input matrices (B) and feedforward (D) are equivalent. The final state space representation of the velocity transfer function is given in Equation 4.6. The pole frequency \( A_V \) tuning is discussed in Section 4.5.2. The final pole frequency in
A = \begin{bmatrix} -W_A \\ \end{bmatrix}, \quad B = \begin{bmatrix} W_A \\ \end{bmatrix} \\
C = \begin{bmatrix} -W_A \\ \end{bmatrix}, \quad D = \begin{bmatrix} W_A \\ \end{bmatrix} \\
(4.15)

x = \frac{A_{AM}}{s}, \quad y = A_{AM}, \quad u = V_M

4.4.2.1 Acceleration Transfer Function - Alternate Derivation

An alternative method of creating the acceleration transfer function (from position input) is as follows.

Equation 4.16 shows the 1st order transfer function relationship of measured velocity $V_M$ to measured acceleration $A_M$. $H_A$ is the acceleration transfer function, and $W_A$ is the acceleration transfer function pole.

\[
A_M = sV_M H_A = sV_M \frac{W_A}{s + W_A}
\]

\[
A_M(s + W_A) = sV_M W_A
\]

\[
sA_M = sV_M W_A - A_M W_A \quad (4.16)
\]

Substituting for $(sV_M)$ from Equation 4.9 into Equation 4.16 will give the following
Equation 4.17.

\[ sA_M = sV_M W_A - A_M W_A \]
\[ = -W_A W_V W_P Z_M + W_A W_V W_P Z_C - W_A W_V V_M - W_A A_M \] (4.17)

Equation 4.17 is now is an easily convertible to state space format, realized in Equation 4.18.

\[
A = \begin{bmatrix} -W_A \end{bmatrix}, \quad B = \begin{bmatrix} -W_A W_V W_P & W_A W_V W_P & W_A W_V \end{bmatrix}, \quad C = \begin{bmatrix} 1 \end{bmatrix}, \quad D = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}
\] (4.18)

\[
x = A_M, \quad y = A_M, \quad u = \begin{bmatrix} Z_M \\ Z_C \\ V_M \end{bmatrix}
\]

4.4.3 Position Transfer Function

The position transfer function dictates the relationship between the LTB commanded position, \( Z_C \), and the measured buoy position, \( Z_M \). The input and output signal locations of this transfer function are shown in the LTB Signal Flow Block Diagram of Figure 4.6.
Figure 4.6: LTB Position Signal Flow Block Diagram

\[ \text{P} = \text{Position Signal Path Start}, \quad \text{P} = \text{Position Signal Path End} \]

### 4.4.3.1 System Response Data Capture

To create the transfer function for position, the LTB was commanded to move in a sinusoidal position path for a variety of periods. The commanded periods and position response information reported back on the LTB position signal were recorded concurrently for analysis and construction of the transfer function.
The periods of the sinusoidal commanded position path were chosen based on typical operating periods of the machine. The amplitude of the sinusoid was enough to displace the buoy, but kept to a minimum height to allow higher frequency (slower period) testing.

To characterize the LTB position transfer function by means of a position profile, 40 different periods were used. The slowest period was 10 seconds, while the quickest period was 0.25 seconds. The sweep step size between periods was 0.25 seconds.

Commanded amplitude of the test sinusoids were 0.025 meters, with zero offset from the LTB zero homing position. The LTB zero homing position did put the 0.025 meter amplitude sinusoidal movement region in a position of motor active component interaction, specifically generator magnets and generator back-iron. The L-10 generator was unloaded for this position characterization, so electrical load did not play a part in the mechanical loading of the LTB.

Forty different sinusoidal position commands were sent to the LTB one at a time. An individual oscillatory period was selected and allowed to continually oscillate for approximately 10 cycles before beginning of measurement occurred. Measurement duration was 20 seconds for all data sets taken, with position sampling rate of the external control computer set to 10,000 samples per second.

The final position transfer function data set consists of 40 commanded position sinusoids (sent from the control computer) and the corresponding 40 measured position sinusoids reported back from the LTB Rexroth motor drive (received by the control computer).

The Simulink model, ControlDesk interface, and associated MATLAB code for
the commanded position characterization can be found in Appendix Section .3.

Examples of the slowest (0.1 Hz) and fastest (4 Hz) sinusoidal oscillating position data sets are respectively shown in Figures 4.7 and 4.8.

Figure 4.7: Slowest Commanded Position, 10 Sec. Period, 0.025 M Amplitude
The forty oscillatory data sets were then analyzed all together for construction of a position-to-position Bode plot as is explained in the next section.

4.4.3.2 Bode Plot Construction

Two methods were used to construct two Bode plots of the Commanded Position to Reported Position of the LTB, one which analyzed the time domain sinusoidal data directly and one which analyzed the fast fourier transforms (FFTs) of the sinusoidal data.

The first method of Bode plot construction was time domain analysis. In looking at Figure 4.7, the distance between the positive slopes of the sinusoids and the distance between the negative slopes of sinusoids is different. This means that the Bode plot...
phase delay will be different for the LTB depending on its direction of motion (up or down). To show the range of possible operations on the Bode plot, two magnitude lines were constructed, one based on the positive sinusoid peaks, and one based on the negative sinusoid peaks. Also, for the phase plot, the phase angle at the positive-slope zero crossings was calculated, along with the phase angle at the negative-slope zero crossing locations. The Bode plot for this analytical method can be seen in Figure 4.9.

Figure 4.9: Position Transfer Function Bode Plot, from Peak and Zero Crossing Comparisons

The second method of Bode plot construction is through use of frequency domain analysis with the Fast Fourier Transform (FFT)[22]. Eighty FFTs were calculated
from the 40 commanded-position and 40 reported-position sinusoidal data sets. The peak values from each of these data set (commanded-to-reported) was used to determine both the magnitude and phase for each of the 40 sinusoidal frequencies used to characterize the LTB position transfer function.

The calculation for the Bode magnitude, which was calculated for each of the 40 individual sinusoids, can be seen in Equation 4.19. In Equation 4.19, \( M_{dB} \) is the magnitude in decibels of the data set being analyzed. \( |R_{MAX}| \) is the peak absolute value of the FFT from the reported (or Rexroth drive) signal of LTB measured buoy position. \( |C_{MAX}| \) is the peak absolute value of the FFT from the commanded (or control computer) signal.

\[
M_{dB} = 20 \log_{10}\left(\frac{|R_{MAX}|}{|C_{MAX}|}\right) \tag{4.19}
\]

The calculation of the Bode phase angles (in degrees), which was calculated for each of the 40 individual sinusoids, can be seen in Equation 4.20. In Equation 4.20, \( A_{deg} \) is the phase angle in degrees for the data set being analyzed. \( \angle(C_{MAX}) \) is the angle of the commanded (or control) sinusoid FFTs peak complex value. \( \angle(R_{MAX}) \) is the angle of the reported (or Rexroth) sinusoid FFTs peak complex value.

\[
A_{deg} = \left(-\angle(C_{MAX}) - \angle(R_{MAX})\right) \frac{180}{\pi} \tag{4.20}
\]

With 40 magnitude and 40 phase points calculated, the final FFT derived Bode plot can be seen in Figure 4.10. Each of the 40 circles on the magnitude and phase lines correspond with the 40 different periods which the LTB was sinusoidally commanded to operate at.
Figure 4.10: Position Transfer Function Bode Plot, from FFT Analysis of Sinusoids

The FFT derived Bode plot was overlaid with the direct time domain analysis Bode plot in Figure 4.11 for comparison purposes.
The FFT derived magnitude and phase lines are essentially centered between the other magnitude and phase lines. This shows that the FFT derived magnitude and phase values will give the average response to the LTB system under all operating conditions experienced during the sinusoidal position testing at 40 different frequencies. For this reason, the FFT derived Bode plot will be used as the comparison point for deriving the actual position transfer function.
4.4.3.3 Fitting 1st Order Transfer Function to Bode Plot

The position transfer function must be fit to the Bode plot of Figure 4.10. There are multiple ways of fitting this transfer function, such as analyzing the Bode plots by direct inspection, or sweeping through a preselected range of pole/zero frequencies and comparing these through mean squared error method to the original FFT derived Bode plot.

Using the direct inspection method on the Bode plot of Figure 4.10, the -3 dB frequency on the gain plot is found to occur at 0.56 Hz. The phase plot’s 45 degree crossing point occurs at 0.79 Hz. The -3 dB frequency and 45 degree frequency indicate the points which a 1st order transfer function should pass through. To average the gain and phase 1st order transfer function pole frequencies together, a frequency of 0.675 Hz served as a good direct inspection 1st order transfer function pole frequency, although the system is obviously more complex than a 1st order transfer function could express.

An improvement on the direct inspection method is to use Mean Squared Error (MSE) analysis to determine the best 1st order pole frequency. MSE analysis gives a means of comparing the errors that different transfer functions produce in reference to the LTB position Bode plot of 4.10. The errors from these different transfer functions can then be compared together, to find the transfer function which gives the smallest error. The MSE equation is given in Equation 4.21.

$$E = \frac{1}{N} \sum_{i=1}^{N} (P_{1st} - P_{FFT})^2$$

(4.21)
Using Equation 4.21 to perform MSE analysis, a sweep of different pole frequencies was performed from 0.01 Hz to 2.0 Hz in 0.01 Hz steps. These pole frequencies were used in a 1st order transfer function which was compared to the original FFT Bode plot. The MSE analysis was performed on both the magnitude and phase plots. The results can be seen in Figure 4.12. The MSE analysis indicated that a 1st order transfer function with a pole frequency of 0.39 Hz best matched the magnitude FFT Bode plot, while a pole frequency of 0.79 Hz best matched the phase FFT Bode plot.

![MSE Magnitude, Minimum Value at \(\omega_c = 0.39\)](image1)

![MSE Phase, Minimum Value at \(\omega_c = 0.79\)](image2)

Figure 4.12: Commanded to Reported Position Mean Squared Error Analysis of Bode Plot

With the MSE best fit pole frequencies being 0.39 Hz for the magnitude plot, and 0.79 Hz for the phase plot, another sweep was performed from 0.4 Hz to 0.8 Hz in
0.1 Hz steps. This was for final visual comparative analysis to choose which 1st order transfer function pole frequency best fits the LTB position response. This sweep from 0.4 Hz to 0.8 Hz can be seen in Figure 4.13.

![Gain Plot for LTB w/ L-10 Position](image1)

The final 1st order pole frequency chosen from Figure 4.13 was 0.8 Hz, in which $W_C$ is equal to $W_P$. This resulted because of excellent matching from 0.1 Hz to 1 Hz of the phase plot, and minimal divergence from the gain plot. It should be

![Phase Plot for LTB w/ L-10 Position](image2)

Figure 4.13: Position Transfer Function Bode Plot with $\omega_C$ Values from 0.4 to 0.8

The final 1st order pole frequency chosen from Figure 4.13 was 0.8 Hz, in which $W_C$ is equal to $W_P$. This resulted because of excellent matching from 0.1 Hz to 1 Hz of the phase plot, and minimal divergence from the gain plot. It should be
pointed out that this 1st order transfer function does not match the LTB Bode plot above 1 Hz, but that the LTB will not (and should not) be operated above the 1 Hz frequency when simulating a buoy in the ocean. The 1st order approximate transfer function is given in Equation 4.22, and its associated state space representation given in Equation 4.23. The final (tuned) pole frequency used in the Kalman filter, in radians per second, is $W_P = 5.35$, which is fairly close to the originally chosen 0.8 Hz.

\[ H_P = \frac{\omega_P}{s + \omega_P} \]  
\[ (4.22) \]

\[
\begin{align*}
A &= \begin{bmatrix} -\omega_P \end{bmatrix}, & B &= \begin{bmatrix} \omega_P \end{bmatrix} \\
C &= \begin{bmatrix} 1 \end{bmatrix}, & D &= \begin{bmatrix} 0 \end{bmatrix} \\
x &= Z_M, & y &= Z_M, & u &= Z_C \\
\end{align*}
\]  
\[ (4.23) \]

4.4.3.4 Fitting 4th Order Transfer Function to Bode Plot

In looking at the Bode plot of Figure 4.13, the LTB magnitude diverges below all of the 1st order transfer function estimates. This indicates a higher pole count system. The magnitude drop from 3 Hz to 4 Hz drops 6.27 dB. This translates to a 63 dB per decade drop in this range of the Bode plot, which would indicate, at a minimum, a three pole transfer function. The phase plot, however, was not falling as fast as would be expected for a 3rd order transfer function. This indicates the presence of zeros in the transfer function.
To simplify fitting the transfer function to the LTB Bode plot, a MATLAB tool was used called ‘System Identification Tool’ (also known as ‘ident’), which allows quick fitting of transfer functions to frequency or time domain data. LTB frequency data was used for fitting a 4th order transfer function with this tool. The system is 4th order because it fit the original LTB Bode plot data better than lower order systems with negligible improvement for higher than 4th order transfer functions. The resulting Bode plot of this best fit 4th order system can be seen in Figure 4.14, and its associated state space representation in Equation 4.24. For Equation 4.24, \( Z_C \) is the commanded position, while \( Z_M \) is the measured position output.
Figure 4.14: Position Transfer Function (4th order) Bode Plot from MATLAB System Identification Tool
4.4.4 Force Transfer Function Characterization

The force transfer function dictates the relationship between the force experienced by the LTB load cells and the force signal output from the LTB. The signal flow path taken into consideration can be seen in Figure 4.15.
4.4.4.1 Load Cell Resistive Simplification

The signal path was not able to be characterized through the entire chain of actual force input to electrical force signal output because technology was not available at the WESRF that would allow a sinusoidal or step characterization of the load cells themselves.
Due to the load cell not being easily characterizable, a simplification of characterization was made. Since the load cell sensor itself is a resistive strain gauge (as discussed in Section 3.1.1), it was assumed that the resistive sensor was not a frequency dependent device, and therefore, did not affect the frequency response of the system. This assumption is not really true of course, because the load cell itself (as shown in Figure 3.1) has associated mechanical properties such as damping and a spring constant. ‘Not really true’ is the reason for the word approximation, which led to characterization of the load cell signal path without the load cell itself.

With the above assumption in place, the expected electrical output of the load cell was calculated based upon an expected force value received by the load cell. This calculated electrical output was created and sent through the remainder of the force signal path, namely the signal path shown in Figure 4.15.

4.4.4.2 Step Response Analysis

To obtain the force signal path transfer function, characterization occurred through the use of a step input response through the system. The idealized step response was composed of infinite sinusoids, effectively meaning that a step response tested the system at all possible frequencies (from zero to infinity).

To determine the appropriate differential step voltage, the load cell capacity, excitation voltage, and rated output voltage must be known. Calculation of the appropriate step voltages for the left and right load cells are shown in Equations 4.25 and

**Left Load Cell (5000 lb Capacity)**

S/N: 263950A  Manufacturer: Interface Force Inc.

**Excitation Voltage:** $10V_{DC}$  **Rated Output:** $2.048 \frac{mV}{V}$

For a 5000 lb load, the output should be as follows:

$$V_{LLC} = 2.048 \frac{mV}{V} \times 10V_{DC} = 20.48mV$$  \hspace{1cm} (4.25)

**Right Load Cell (5000 lb Capacity)**

S/N: 263914A  Manufacturer: Interface Force Inc.

**Excitation Voltage:** $10V_{DC}$  **Rated Output:** $2.041 \frac{mV}{V}$

For a 5000 lb load, the output should be as follows:

$$V_{LLC} = 2.041 \frac{mV}{V} \times 10V_{DC} = 20.41mV$$  \hspace{1cm} (4.26)

Since the voltage output from the load cell was a differential voltage relative to the load cell, two Control Computer dSPACE analog output signals were used to create the differential voltage. The analog output signals were each biased to 5 volts so that the differential voltage between each of them is 0 volts. For characterizing the Left Load Cell, the differential step voltage of $20.48mV$ (Equation 4.25) was applied mathematically to the 5 volt signal lines.

To apply this voltage from dSPACE as a differential output, the voltage was
divided by two (10.24\text{mV} for the Left Load Cell), and separately added and subtracted to a $+5$ volt offset voltage. This created the differential voltage signal on the load cell output pins (+) and (−), and provided a common mode voltage of $+5$ volts. This mathematical function is described in block diagram format in Figure 4.16.

![Load Cell Step Response Signal Creation Block Diagram](image)

Figure 4.16: Load Cell Step Response Signal Creation Block Diagram

With the cable connected to each of the load cells (separate tests for each), the estimated load cell step response was fed into the LTB force signal path and the output of that path was recorded on the external control computer. Step response data was recorded for 10 seconds, and in that time frame, 12 arbitrarily located step response events occurred. Table 1 of the Appendix shows the pin out for the cable connecting the dSPACE Control Computer to the LTB for simulated step response of the load cells.
A closeup of the normalized left load cell signal path step input and step response output is shown in Figure 4.18. The normalized right load cell step response closely resembled the left load cell step response, as seen in Figure 4.19.
Figure 4.18: Closeup of a Left Load Cell Step Response Occurring
4.4.4.3 Transfer Function Fit to Step Response Results

The transfer function was fit to the output signal responses of the load cells as shown in Figures 4.18 and 4.19. The transfer function, as shown in Equation 4.27, related the actual force a buoy is experiencing, $F$, to the measured force presented on the Left and Right Load Cell feedback signal lines, $F_M$. Upon inspecting Figures 4.18 and 4.19, the load cell step response is that of an under-damped system because of the exponentially damped sinusoidal oscillations. A 2nd order transfer function was calculated and fitted to this step response.

The standard 2nd order transfer function frequency domain equation is given in
Equation 4.28, with $\omega_n$ being the natural frequency, and $\zeta$ being the damping ratio. A small time delay was observed between the step time and output response. To account for this, the ‘Time Shift Theorem’ equation, Equation 4.29 was used, in which $T_D$ is the system time delay. The combination of these equations will give the transfer function solution to the load cell signal path step response, Equation 4.30.

$$F_M = H_F(s)F$$  
(4.27)

$$T(s) = \frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2}$$  
(4.28)

$$H(s) = e^{-sT_D}T(s)$$  
(4.29)

$$H_F(s) = \frac{F_M}{F} = e^{-sT_D} \frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2}$$  
(4.30)

The first constant to be calculated was the natural frequency, $\omega_n$. Natural frequency was derived for both Figures 4.18 and 4.19 based on the inverse period of the exponentially damped oscillation. For both figures, $\omega_n$ is found to be 3927 radians per second.

To calculate the damping ratio, $\zeta$, the Mean Squared Error (MSE) was calculated (Equation 4.21) for different $\zeta$ values. Because the system was underdamped (exponentially damped oscillations), the $\zeta$ value should be between zero and 1. The $\zeta$ value that gave the lowest MSE for the Left Load Cell was 0.29. For the Right Load Cell, the lowest MSE occured at $\zeta$ equal to 0.36. An average of these two $\zeta$ values, 0.32 was used in the final force transfer function.

The final value to be calculated was the small time delay between the input step
and step response. This can be calculated by inspection of the Figures 4.18 and 4.19. The delay was the same for both of these, with the time delay value for $T_D$ equal to 0.0001 seconds.

The force transfer function shown in Equation 4.30 was overlaid onto the normalized original step response plots. The Left Load Cell comparison can be seen in Figure 4.20, while the Right Load Cell can be seen in Figure 4.21.

Figure 4.20: Force Transfer Function Matched to Left Load Signal Path Step Response
Figures 4.20 and 4.21 show the effect of the damping ratio value used (0.32) being slightly more than the Left Load Cell value (0.29) and slightly less than the Right Load Cell value (0.36). The Left Load Cell exponentially damped oscillation had a slightly larger amplitude (less damping) than the transfer function, while the Right Load Cell exponentially damped oscillation had a slightly smaller amplitude (more damping) than the transfer function. This small difference should be considered insignificant to the final Kalman filter, although a more complex Kalman filter may be possible which uses both damping ratios.

The force transfer function, originally represented in Equation 4.30 is now represented in state space form as seen in Equation 4.31. The constants derived above
such as $T_D$, $\omega_n$, and $\zeta$ must be substituted in for Kalman filter creation.

\[
A = \begin{bmatrix}
-2\zeta\omega_n & \omega^2_n & \omega^2_n \\
1 & 0 & 0 \\
0 & 0 & 0
\end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}
\]

\[
C = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix}, \quad D = \begin{bmatrix} 0 \end{bmatrix}
\]  

(4.31)

\[
x = \begin{bmatrix}
\frac{d}{dt}F_M \\
F_M \\
F
\end{bmatrix}, \quad y = \begin{bmatrix} F_M \end{bmatrix}, \quad u = \begin{bmatrix} W_{FLTB} \end{bmatrix}
\]

\[
InputDelay = T_D
\]  

(4.32)

Note the variable $W_{FLTB}$ in Equation 4.31. It is a normally distributed stochastic variable, which accounts for the lack of a commanded variable directly affecting the force changes being experienced by the buoy in the LTB. An example of a commanded variable would be the commanded position variable, which directly affects the position, velocity, and acceleration measurements. There is no direct relationship between commanded position and force, hence the need for the normally distributed stochastic variable input to the force transfer function state space equations.

The $InputDelay$ variable underwent a Padé approximation to convert it from a fixed time delay into a 1st order rational linear time invariant system (a transfer function). For reference (as the Kalman filter has implemented), a 1st order Padé
approximation of a time delay $T_D = 0.0001 \text{seconds}$ is shown in the Equation 4.33.

$$T_D(s) = \frac{-s + 20000}{s + 20000}$$ (4.33)

4.5 Kalman Filter Assembly

Construction of the Kalman filter required three component: a single, assembled, state space representation of the entire system, the Process Noise Covariance Matrix $Q_N$, and the sensor noise covariance matrix, $R_N$. The variance of the process noise acts as a knob for tuning the Kalman filter estimates. This will be discussed more in Section 4.5.2.

4.5.1 Sub-System Inner-Connection

To construct a single state space representation of the LTB, the transfer functions (represented in state space form) were combined together. This was performed with the MATLAB `connect` command, which allows for name-based interconnection of the state space inputs and outputs.

Once all subsystems of the single system were inner-connected, the Padé approximation was performed, removing the input time delay as a constant time value and adding the input time delay as a rational LTI function to the state space representation of the system. The Padé approximation state can be seen as variable $X$ of the state variables. The final numerical state space representation of the LTB’s position, velocity, acceleration, and force transfer function is given below in Equation 4.34. Note that this is the final SS representation after Kalman filter tuning occurred,
although it is not the Kalman filter itself (Section 4.5.2).

$$A = \begin{bmatrix}
-5.35 & 0 & 0 & 0 & 0 & 0 & 0 \\
31.42 & -31.42 & 0 & 0 & 0 & 0 & 0 \\
493.5 & -493.5 & -15.71 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & -2513 & -1.542e + 007 & 1.542e + 007 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 156.3 \\
0 & 0 & 0 & 0 & 0 & 0 & -2e + 004
\end{bmatrix}$$

$$B = \begin{bmatrix}
5.35 & 0 \\
0 & 0 \\
0 & 0 \\
0 & 0 \\
0 & -1 \\
0 & 256
\end{bmatrix}$$

$$C = \begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 \\
31.42 & -31.42 & 0 & 0 & 0 & 0 \\
493.5 & -493.5 & -15.71 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0
\end{bmatrix}$$

$$D = [0]$$

$$x = \begin{bmatrix}
Zm \\
intVam \\
intAam \\
dFltbm \\
Fltbm \\
Fltb \\
X
\end{bmatrix}, \quad y = \begin{bmatrix}
Zm \\
Vam \\
Aam \\
Fltbm
\end{bmatrix}, \quad u = \begin{bmatrix}
Zc \\
wFltb
\end{bmatrix}$$ (4.34)
4.5.2 Tuning

Tuning the Kalman filter involved tuning transfer function pole frequencies $\omega_P$, $\omega_V$, $\omega_A$, and the variance value of the process noise $Q_N$. These final tuned values were used to construct the LTB system matrix which is given in the previous section, Section 4.5.1.

Tuning was performed by actuating the LTB with a 0.05 meter amplitude sine wave which had a 5 second period. The 0.05 meter amplitude was chosen because that range of motion will allow a complete cycle of generator cogging forces to occur. While the Kalman filter was optimized at this particular sinusoidal amplitude and frequency, there may different frequencies and gains which would provide more idealized estimates of the position, velocity, acceleration, and force.

4.5.2.1 Position Transfer Function Tuning - $\omega_P$

The first pole frequency to be tuned was $\omega_P$. This was tuned first because velocity and acceleration are a dependent of it, while force is not directly dependent on it. The final 5.35 rad/sec were relatively close to the initially chosen value of 0.8 Hz (5.03 rad/sec).

To tune the position pole $\omega_P$, the raw position feedback signal coming from the LTB was compared directly with the Kalman estimate of position. The test sine wave zero-crossing points were visually observed for phase delay (lead or lag) which was minimized.
4.5.2.2 Velocity Transfer Function Tuning - $\omega_V$

The second pole to be tuned was the velocity pole, $\omega_V$. This was tuned after the position pole because of velocity’s dependence on the position measurements from the LTB. The final pole frequency chosen was $10\pi \frac{rad}{sec}$. This pole frequency essentially acts as a low pass filter, filtering out higher velocity signals which are more likely to be system noise than actual LTB velocity changes.

To tune the velocity pole $\omega_V$, the raw velocity feedback signal coming from the LTB was compared directly with the Kalman estimate of velocity. The raw velocity signal had some noise associated with being the time derivative of the measured position signal. This noise was minimized by lowering the pole frequency as much as possible without causing any phase delay (lead or lag) between the raw velocity signal and the Kalman estimated signal.

4.5.2.3 Acceleration Transfer Function Tuning - $\omega_A$

The third pole to be tuned was the acceleration pole, $\omega_A$. This was tuned after the velocity pole because of acceleration’s dependence on both the position and velocity measurements from the LTB. The final pole frequency chosen was $5\pi \frac{rad}{sec}$. This pole frequency essentially acts as a low pass filter, filtering out higher acceleration signals, which are more likely to be system noise than actual LTB velocity changes. There is a significant amount of acceleration noise due to 2nd order time derivatives of the position signal to calculate acceleration.

To tune the acceleration pole $\omega_A$, the raw acceleration feedback signal coming from
the LTB was compared directly with the Kalman estimate of acceleration. The raw acceleration signal had a great deal of noise which was clearly filtered out. The filter frequency was adjusted to prevent acceleration signal jitter, and prevent phase delays (lead or lag) between the raw acceleration signal and the Kalman filter estimate of acceleration.

4.5.2.4 Force Transfer Function Tuning - $Q_N$

The force transfer function is tuned with the variance of process noise - $Q_N$. This is because the measured force is not directly dependent on any of the system inputs, such as position, but is dependent on the integration of process noise from the system to change the resulting force estimate.

A larger variance of the process noise, $Q_N$ will allow the force estimate to change more quickly. However, this will also allow more noise to appear on the force measurement estimate. A smaller $Q_N$ will filter out more of the undesired process noise, but if $Q_N$ is made too small, then significant phase delay can result in the changing of the force signal estimate as the actual force changes in the machine.

The final $Q_N$ value used was 10, which allowed for quick response of the force signal while minimizing phase delay and the small amount of noise on the raw force feedback signal.
4.5.3 LTB Kalman Filter Matrices

The three pieces necessary to construct a Kalman Estimator were derived and ready for construction of the Kalman filter.

The Kalman Filter was constructed in MATLAB using the ‘kalman’ command. It is shown in Equation 4.35. The Kalman filter has one known input, commanded position $Z_C$, and four measured inputs, measured position $Z_M$, measured velocity $V_M$, measured acceleration $A_M$, and measured force $F_M$. The estimated outputs are estimated measured position $Z_{Me}$, estimated measured velocity $V_{Me}$, estimated measured acceleration $A_{Me}$, and estimated measured force $F_{Me}$. Additionally, there
are seven state estimated outputs which are not used at this time.

\[
A = \begin{bmatrix}
-5.35 & 0 & 0 & 0 & 0 & 0 & 0 \\
31.42 & -31.42 & 0 & 0 & 0 & 0 & 0 \\
493.5 & -493.5 & -15.71 & 0 & 0 & 0 & 0 \\
3.725e7 & -3.587e+7 & 2.745e+5 & -2513 & -1.936e+7 & 1.542e+7 & 0 \\
2.653e+4 & -2.556e+4 & 195.5 & 1 & -2808 & 0 & 0 \\
2.656e+04 & -2.552e+4 & 195.3 & 0 & -2804 & 0 & 156.3 \\
-1.945e-8 & 1.873e-8 & -1.433e-10 & 0 & 2.058e-9 & 0 & -2e+4
\end{bmatrix}
\]

\[
B = \begin{bmatrix}
5.35 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 \\
0 & -1.371e+006 & -1.416e+006 & 1.748e+004 & 3.941e+006 \\
0 & -976.4 & -1009 & 12.45 & 2808 \\
0 & -975.1 & -1008 & 12.43 & 2804 \\
0 & 7.158e-010 & 7.397e-010 & -9.126e-012 & -2.058e-009
\end{bmatrix}
\] (4.35)

\[
C = \begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 \\
31.42 & -31.42 & 0 & 0 & 0 & 0 & 0 \\
493.5 & -493.5 & -15.71 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1
\end{bmatrix} , \quad D = [0]
\] (4.36)
4.6 Kalman Filter Results

4.6.1 Steady State Error Analysis

As a means of evaluating the position, velocity, acceleration, and force signals for improvement, the variance of each signal was calculated with the LTB in a static (non-moving) state, at a homed (zeroed) position. These signals were recorded for 60 seconds both in raw (unfiltered) and Kalman estimated forms for comparison purposes. The recorded signals are shown in Figure 4.22.
Variance and average calculations are given in Table 4.11 for each of the signals measured.

The estimated position signal $P_E$ shown in Figure 4.22 has a slight estimated position offset. This estimated position offset is due to noise on the commanded signal line, which is shown when looking at the average of the raw commanded signal data in Table 4.11. This error would be removed with the combining of the two external control computers (as mentioned in Appendix Section 2.2). This would bring the commanded position signal from the noisy analog domain into the digital domain,
removing any noise.

Table 4.11: Variance and Average of Signals in Static State

<table>
<thead>
<tr>
<th>Signal</th>
<th>Variance</th>
<th>Average</th>
</tr>
</thead>
<tbody>
<tr>
<td>Commanded Position $P_C$</td>
<td>6.9134e-008</td>
<td>-7.9860e-004</td>
</tr>
<tr>
<td>Raw Position $P_R$</td>
<td>1.6822e-006</td>
<td>-2.3362e-004</td>
</tr>
<tr>
<td>Estimated Position $P_E$</td>
<td>2.8238e-011</td>
<td>-8.0208e-004</td>
</tr>
<tr>
<td>Raw Velocity $V_R$</td>
<td>4.0394e-006</td>
<td>-0.0032</td>
</tr>
<tr>
<td>Estimated Velocity $V_E$</td>
<td>3.3249e-009</td>
<td>-2.2010e-007</td>
</tr>
<tr>
<td>Raw Acceleration $A_R$</td>
<td>.0307</td>
<td>0.0107</td>
</tr>
<tr>
<td>Estimated Acceleration $A_E$</td>
<td>6.9337e-007</td>
<td>-9.2217e-007</td>
</tr>
<tr>
<td>Raw Force $F_R$</td>
<td>8.2670e+003</td>
<td>3.7904e+003</td>
</tr>
<tr>
<td>Estimated Force $F_E$</td>
<td>8.0325e+003</td>
<td>3.7904e+003</td>
</tr>
</tbody>
</table>

The ratios of raw feedback signal variance divided by estimated feedback signal variance can be seen in Table 4.12. The percentage improvements are given in the right column of this table.

Table 4.12: Static State Signal Improvements

<table>
<thead>
<tr>
<th>Signal Ratio</th>
<th>Improvement Ratio</th>
<th>Percentage Improvement</th>
</tr>
</thead>
<tbody>
<tr>
<td>$P_R/P_E$</td>
<td>5.9574e+004</td>
<td>5,957,300%</td>
</tr>
<tr>
<td>$V_R/V_E$</td>
<td>1.2149e+003</td>
<td>121,390%</td>
</tr>
<tr>
<td>$A_R/A_E$</td>
<td>4.4322e+004</td>
<td>4,432,100%</td>
</tr>
<tr>
<td>$F_R/F_E$</td>
<td>1.0292</td>
<td>2.92%</td>
</tr>
</tbody>
</table>
The position, velocity, and acceleration signals all experience marked improvement in variance reduction, with the position signal variance reduced by almost 6 million percent. The acceleration signal variance, which has been the cause of the greatest noise, was reduced by almost 4.5 million percent.

The force improvement is minimal, with almost no change in variance between the raw and estimated force signals. The force signal is considered reasonably smooth as compared to the other signals, and too much smoothing of the force signal by increasing $Q_N$ of the Kalman filter would result in lost force measurement and phase delay in force measurement. Minimal filtration of the force signal is desired to allow high frequency generator cogging forces (relative to generator oscillatory frequency) to appear on the force signal estimate.

4.6.2 Simulated Ocean Wave Data

Four simulated ocean wave data files[23] were used to simulate actual ocean excitation of the L-10 buoy inside the LTB. The ocean wave data was commanded from the external force control computer to the LTB, while a second external control computer was used to perform Kalman estimations of the LTB feedback signals. Figures 4.23, 4.24, 4.25, 4.26, 4.27, 4.28, 4.29, and 4.30 show the position, velocity, acceleration, and force that the buoy experienced while in the LTB. The figures are all taken from the same 60 second recording of L-10 buoy response in the LTB to the wave data set. The wave data file used for LTB excitation is named '1p88m_7p5sec.mat' which indicates that the largest waves experienced were 1.88 meters in height (peak to peak),
and had a minimum period of 7.5 seconds.

4.6.2.1 Position

Commanded position, raw measured position, and Kalman filtered position data can be seen in Figures 4.23 and 4.24.

![Wave Follower - Position: Commanded, Raw, & Kalman](image)

Figure 4.23: Position Command, Raw Response, and Estimation of Ocean Waves

The position data closeup, Figure 4.24 shows the expected phase delay between commanded position and measured position. The Kalman estimates of position line up well with the raw position signal along the areas of constant velocity, such as near
the zero crossing points in this particular plot.

The plot areas which experience a changing velocity, however, such as at the peaks of the waves, do not show a good estimated match of the raw position signal. Better estimates of the wave peaks may be possible with a more accurate position transfer function at higher frequencies. A higher order transfer function fitted more closely to the position Bode plot may aid in improving the misalignment of the Kalman estimation.

![Wave Follower - Position: Commanded, Raw, & Kalman](image)

Figure 4.24: Closeup of Position Command, Raw Response, and Estimation of Ocean Waves
4.6.2.2 Velocity

The raw velocity feedback signal along with its Kalman estimated signal can be seen in Figure 4.25.

Reasonably good signal matching occurs between the raw velocity and the Kalman filter estimate, particularly along the velocity line’s constant acceleration areas. There is some small mismatch when the velocity changes rapidly, such as at the valley which occurs at approximately 22 seconds on Figure 4.26. This slight lead in the Kalman estimation velocity signal may be eliminated by tuning the velocity pole frequency,
$W_V$, to a higher value to allow it to change more rapidly.

Figure 4.26: Closeup of Velocity Raw Response and Estimation of Ocean Waves

4.6.2.3 Acceleration

The dynamic acceleration signal estimation shows the most marked improvement of all dynamic feedback signal estimates, as is shown in Figure 4.27.
Figure 4.27: Acceleration Raw Response and Estimation of Ocean Waves

Visual inspection of the 10 second acceleration plot shown in Figure 4.28 shows that the Kalman estimated acceleration has a slight lead to what appears to be the average point of the raw acceleration signal. This estimation signal lead can be removed by reducing the acceleration pole frequency $W_A$.

The noisiest section of the acceleration plot occurs between 24 and 27 seconds. This corresponds with the magnetic cogging forces seen in the load cell force Figure 4.30 in the same time frame. This increased noise indicates that the acceleration may be changing on a small level in the acceleration plot where the large block of noise
occurs, of which noise is amplified in the 2nd order time derivative of position to calculate the system acceleration. This could be achieved by increasing the acceleration pole frequency, $W_A$, and should be determined when the entire Linear Quadratic Gaussian Controller is tested.

Figure 4.28: Closeup of Acceleration Raw Response and Estimation of Ocean Waves

4.6.2.4 Force

The dynamic force signal estimation in Figure 4.27 shows minimal difference from the dynamic raw force signal. Kalman filter tuning of the force signal was performed
by changing $Q_N$, the variance of the process noise. The figure of the force signal is believed to be of reasonable accuracy and low noise because of the high speed response and resistive sensor qualities of the load cells.

![Wave Follower - Force: Raw & Kalman](image)

Figure 4.29: Force Raw Response and Estimation of Ocean Waves
Note: ‘Raw’ Signal can not be seen because of Kalman filter almost perfectly matching raw signal.

Generator cogging, which occurs between the float magnets and generator back-iron is clearly present between 24 and 26 seconds in Figure 4.28. Decreasing $Q_N$ will effectively smooth out the estimated force feedback signal, but, this would not be considered an accurate representation of the force feedback signal as it is relatively
noise free when compared to the other feedback signals.

Figure 4.30: Closeup of Force Raw Response and Estimation of Ocean Waves
Note: ‘Raw’ Signal can not be seen because of Kalman filter almost perfectly matching raw signal.
Chapter 5 – Conclusion

This thesis covered two core topics that are necessary for advancement of the LTB for use as an ocean simulator of linear generating devices. To have the LTB accurately simulate an ocean energy generating buoy experiencing actual ocean wave forces, a closed-loop force control scheme featuring properly calibrated and noise free feedback signals must be implemented.

5.1 Load Cell Conclusion

It was shown that the LTB’s load cells do drift slightly from measured values over time, but that WESRF-based load cell calibration will correct this sensor accuracy degradation. While only a partial calibration check was used to determine load cell degradation for this thesis, a full calibration procedure, including LTB hardware changes, is described in this thesis which will allow the calibration procedure to be safer, mechanically simpler, and accurate to industry standards.

Implementation of the full calibration process is possible through creation of two fixed mounting points to the LTB. With the implementation of these fixed mounting points, full capacity force loads may be applied to the LTB load cells in both tension and compression using an S-type force measuring gauge.
5.2 Kalman Filter Conclusion

With proper characterization of the LTB’s position, velocity, and acceleration transfer functions, and knowledge of the statistical noise present on the feedback signal lines, a Kalman filter was constructed that mathematically removes the noise on the feedback signal lines. The Kalman filter is able to do this without adding the phase delay that typically results from low pass filters traditionally used to filter ambient noise.

5.3 Recommended Areas of Future Work

5.3.1 Force Control Completion

To complete LTB force control of buoy devices under test, the closed-loop force control must be completed by designing a Linear Quadratic Optimal Gain $K$, as mentioned in Section 4.1. With Linear Quadratic Optimal Gain $K$, better simulation of ocean wave buoy force interactions is possible.

5.3.2 Acceleration Signal

Further investigation of the acceleration signal accuracy should be investigated in one or both of the following ways.

The Directional Wave Rider, which is an ocean wave capturing device with built-in accelerometer, should be attached to the LTB carriage assembly. This device will then be instructed to record movement of the LTB as if it were out in the ocean.
Previous ocean wave data gathered by the Directional Wave Rider will then be fed into the LTB, commanding it to replicate the ocean waves which the Directional Wave Rider previously experienced. The position, velocity, and acceleration signals originally recorded by the Directional Wave Rider can then be compared to the LTB’s position, velocity, and acceleration signals. This comparison will provide a method of analyzing the LTB’s ability to accurately replicate the Directional Wave Rider’s recorded wave data.

A second suggested method of improving LTB acceleration signal feedback is to attach an accelerometer to the device under test (such as the L-10 generator). This acceleration signal can then be fed back to the external control computer for comparison with the LTB’s 2nd order derivative of position calculated acceleration signal. The accelerometer would provide feedback information directly based upon physical acceleration experienced by the accelerometer sensor, removing the effect of the 2nd order time derivative amplifying position feedback signal noise.

To prepare the LTB for future load cell calibration procedures, hardware preparations should be made so that load cell calibration can quickly and safely be performed when test devices such as the L-10 are removed from the LTB. These hardware suggestions are discussed in Section 3.3 and mentioned in the following list.

- Create fixed point LTB baseplate bracket to be used for load cell tension tests.
- Create removable fixed point steel crossbar at top of LTB to be used for load cell compression tests.
- Create LTB carriage arm central mounting point for load cell suspension arm
5.4 Ocean Energy Conclusion

Access to the wealth of clean renewable ocean energy can only be tapped after solving a number of technological problems. To effectively solve these problems, the scientific method and good engineering practices will be the tools which lead to robust and cost effective solutions. OSU’s Ocean Wave Energy Linear Test Bed is an important tool for characterization and analysis of point absorber linear ocean wave energy generators, and with development of its closed-loop force control, more accurate buoy-wave force interaction simulations will assist in the search for technological solutions that help solve the world’s growing needs for clean renewable energy.
Bibliography


APPENDICES
.1 Load Cell Test Sheet

<table>
<thead>
<tr>
<th>Test Point</th>
<th>Units</th>
<th>LTB Load Cell Units</th>
<th>S-Type Load Cell Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>lbs</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-1000</td>
<td>lbs</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-2000</td>
<td>lbs</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-3000</td>
<td>lbs</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-4000</td>
<td>lbs</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-5000</td>
<td>lbs</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-2000</td>
<td>lbs</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>lbs</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Tension Test Hardware Configuration

<table>
<thead>
<tr>
<th>Test Point</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>lbs</td>
</tr>
<tr>
<td>-1000</td>
<td>lbs</td>
</tr>
<tr>
<td>-2000</td>
<td>lbs</td>
</tr>
<tr>
<td>-3000</td>
<td>lbs</td>
</tr>
<tr>
<td>-4000</td>
<td>lbs</td>
</tr>
<tr>
<td>-5000</td>
<td>lbs</td>
</tr>
<tr>
<td>-2000</td>
<td>lbs</td>
</tr>
<tr>
<td>0</td>
<td>lbs</td>
</tr>
</tbody>
</table>

Compression Test Hardware Configuration

<table>
<thead>
<tr>
<th>Test Point</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>lbs</td>
</tr>
<tr>
<td>1000</td>
<td>lbs</td>
</tr>
<tr>
<td>2000</td>
<td>lbs</td>
</tr>
<tr>
<td>3000</td>
<td>lbs</td>
</tr>
<tr>
<td>4000</td>
<td>lbs</td>
</tr>
<tr>
<td>5000</td>
<td>lbs</td>
</tr>
<tr>
<td>2000</td>
<td>lbs</td>
</tr>
<tr>
<td>0</td>
<td>lbs</td>
</tr>
</tbody>
</table>

Remove All Test Hardware

<table>
<thead>
<tr>
<th>Test Point</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>lbs</td>
</tr>
</tbody>
</table>

Figure 1: Suggested 5000 lb Load Cell Test Sheet
.2 Computer Configurations

.2.1 Hardware Setup for Position Transfer Function Characterization

All input and feedback sinusoids used to create the position transfer function Bode plot were recorded using the external control computer’s dSPACE DSP card and ControlDesk interface.

![Figure 2: Position Transfer Function Computer Configuration](image)

.2.2 Hardware Setup for Force Transfer Function Characterization

All input and feedback step response signals used to create the force transfer function were recorded using the external control computer’s dSPACE DSP card and ControlDesk interface.
2.3 Kalman Filter Tuning and Testing

Two computers were used in the tuning process as is shown in Figure 4. This significantly sped up the tuning process, although some inaccuracy was introduced into the Kalman filter because of noise associated with converting from digital to analog and back to digital when moving the commanded position signal between computers. The noise issue is pointed out in Section 4.6.
Figure 4: Kalman Filter Tuning and Testing Computer Configuration

The reasoning behind the use of two computers has to do with the Kalman Filter matrices being updated continually and loaded onto the dSPACE DSP in the control computer. Each time the dSPACE DSP is updated, it turns off all analog and digital outputs. The loss of output on the digital safety innerconnect between the LTB and the external control computer causes the LTB to fault, requiring the LTB carriage to be re-homed (a time consuming process). -
.3 Position Transfer Function Characterization

.3.1 Simulink Model

Figure 5: Position Transfer Function Simulink Model
.3.2 ControlDesk Interface

![ControlDesk Interface Figure]

Figure 6: Position Transfer Function ControlDesk Interface

.3.3 MATLAB Code for Initialization of Simulink Blocks

```matlab
% Initialization file for LTB Transfer Function Creation of Load Cell
%Signal moving through System
%(from output of load cell ((MIL connector)))
% to
%(BNC Left Load Cell & Right Load Cell) on side
%of control cabinet
%Creation: Chris Haller, 2010 April 5

% 5000 LBS Load-Cells
LeftCell_gain = 5000*4.44822*1.00130;
LeftCell_zero_offset = 5+9.9;
RightCell_gain = 5000*4.44822*0.996359;
```
RightCell_zero_offset = 13.5+6.5;

% 1000 LBS Load-Cells
%LeftCell_gain = 5000*4.44822*1.00130;;
%LeftCell_zero_offset = 5;
%RightCell_gain = 5000*4.44822*0.996359;
%RightCell_zero_offset = 13.5;

% 250 LBS Load-Cells
%LeftCell_gain = 250*4.44822*1.00130;
%LeftCell_zero_offset = 6.5;
%RightCell_gain = 250*4.44822*1.00450;
%RightCell_zero_offset = 6.5;

disp('transfer_function_init.m complete')

.3.4 MATLAB Code for Calculation of Position Bode Plot

%=====================================================================%
% OSU-WESRF LTB Transfer Function Bode Plot Constructor
% Christopher Haller
% Created: 2010 March 11
%=====================================================================%

% Initialization of MATLAB
logspace
% Constructs the transfer function bode plot for the linear test bed
% Input sinusoids are fed into "dynamic force control" input of LTB
% Output sinusoids are read from "Position" output from Rexroth drive
% Sample Rate: 10,000 per second
% Y1 = Rexroth Data
% Y2 = dSPACE Commanded Position
%=====================================================================%

close all;
clear;
clc;

%format compact;
firstMethodBode = 1; %1st method of Bode calculation

%% Load Sinusoidal Data into Dataset Struct variable
for i = 1:40
    filepath = strcat('MATLAB\DATA\', num2str(i), '.mat');
    S = load(filepath);
    names = fieldnames(S);
    Dataset(i,1) = S.(names{1,1});
end

%% Plot waveforms (not needed, for testing only)
% figure(1);
% hold on;
% for i = 40:40
%    plot(Dataset(i).X.Data,Dataset(i).Y(1).Data);
%    plot(Dataset(i).X.Data,Dataset(i).Y(2).Data, '--r');
% end
% legend('got','want');

%% Calculate Frequencies / Time Periods of Datasets
timePERIODS = 10:-0.25:0.25;
frequencies = 1./timePERIODS;

%% Filter Noise on Rexroth signal (Y1)
if (firstMethodBode)
    Dataset_filt = struct('Y1',[]);
    [num, den] = butter(5, 50/5000, 'low');
    for i = 1:40
        Dataset_filt(i).Y1 = filtfilt(num,den,Dataset(i).Y(1).Data);
    end
end

%% Determine zero-crossing points (Determines Indexes)
if (firstMethodBode)
    Dataset_zeros = struct('Y1',[],'Y2',[]);
    for i = 1:40
        %Rexroth (filtered)
        Dataset_zeros(i).Y1 = crossing(Dataset_filt(i).Y1);
        %Commanded
        Dataset_zeros(i).Y2 = crossing(Dataset(i).Y(2).Data);
%% Calculate Phase Difference
if (firstMethodBode)
    for i = 1:40
        IndexDSPACE = Dataset_zeros(i).Y2(1);
        IndexRexroth = Dataset_zeros(i).Y1(1);
        TimeDSPACE = Dataset(i).X.Data(IndexDSPACE);
        TimeRexroth = Dataset(i).X.Data(IndexRexroth);

        %Might be missing 1st zero of commanded data
        % ( > 180 degree phase offset)
        if abs(TimeDSPACE - TimeRexroth) > (0.25*timePERIODS(i))
            %Remove a Rexroth (always lagging) zero
            Dataset_zeros(i).Y1 = Dataset_zeros(i).Y1(2:end);
            % fprintf('n WORKING!!!');
        end

        %Special case for 40.mat file &*(@#*()$&()ˆ@#$ is missing 1st ctrl-0
        if i == 40
            Dataset_zeros(i).Y1 = Dataset_zeros(i).Y1(2:end);
        end

        %Phase Delay Calculation (PHASE DELAY IS IN DEGREES!!)
        %IF Slope Is Positive
        if ( Dataset(i).Y(2).Data(Dataset_zeros(i).Y2(i)-10) < 0 )
            %Phase Delay for Positive Slope
            IndexDSPACE = Dataset_zeros(i).Y2(3);
            IndexRexroth = Dataset_zeros(i).Y1(3);
            TimeDSPACE = Dataset(i).X.Data(IndexDSPACE);
            TimeRexroth = Dataset(i).X.Data(IndexRexroth);
            PhaseSINEpos(i) = (TimeDSPACE - TimeRexroth) ...
            / timePERIODS(i) *360;

            %Phase Delay for Negative Slope
            IndexDSPACE = Dataset_zeros(i).Y2(2);
            IndexRexroth = Dataset_zeros(i).Y1(2);
            TimeDSPACE = Dataset(i).X.Data(IndexDSPACE);
            TimeRexroth = Dataset(i).X.Data(IndexRexroth);
            PhaseSINEneg(i) = (TimeDSPACE - TimeRexroth) ...
            / timePERIODS(i) *360;
        else %SLOPE IS NEGATIVE

        end
    end
end

end
% Phase Delay for Positive Slope
IndexDSPACE = Dataset.zeros(i).Y2(3);
IndexRexroth = Dataset.zeros(i).Y1(3);
TimeDSPACE = Dataset(i).X.Data(IndexDSPACE);
TimeRexroth = Dataset(i).X.Data(IndexRexroth);
PhaseSINEneg(i) = (TimeDSPACE - TimeRexroth) ... /
timePERIODS(i) * 360;

% Phase Delay for Negative Slope
IndexDSPACE = Dataset.zeros(i).Y2(2);
IndexRexroth = Dataset.zeros(i).Y1(2);
TimeDSPACE = Dataset(i).X.Data(IndexDSPACE);
TimeRexroth = Dataset(i).X.Data(IndexRexroth);
PhaseSINEpos(i) = (TimeDSPACE - TimeRexroth) ... /
timePERIODS(i) * 360;
end
end

% PhaseSINEpos
% PhaseSINEneg
end

%% Calculate Gain Magnitude (in dB)
if (firstMethodBode)
for i = 1:40
% Gain of sinewave positive peaks
maxREXROTH = max(Dataset(i).Y(1).Data);
maxDSPACE = max(Dataset(i).Y(2).Data);
MagSINEpos(i) = 20 * log(maxREXROTH/maxDSPACE);

% Gain of sinewave negative peaks
maxREXROTH = max(-Dataset(i).Y(1).Data);
maxDSPACE = max(-Dataset(i).Y(2).Data);
MagSINEneg(i) = 20 * log(maxREXROTH/maxDSPACE);
end
end

%% Extract Sampling Frequency from Each Data Set
for i = 1:40
dataLENGTH(i) = length(Dataset(i).X.Data);
timeFRAME(i) = max(Dataset(i).X.Data) - min(Dataset(i).X.Data);
Fs(i) = round(dataLENGTH(i)/timeFRAME(i)); % Sample Frequency
T(i)=dataLENGTH(i)/Fs(i); % Time Elapsed
end
% Calculate FFT's and Frequencies/Time-Periods of Y2 Datasets
% for i = 1:1
% %Create FFT of data
% FFTY2set(i) = centeredFFT(Dataset(i).Y(2).Data, Fs(i));
% % Determine period of each sine wave
% [valueMAX indexMAX] = max(FFTY2set(i).fftVALS);
% timePERIODS(i) = 1 / abs(FFTY2set(i).freq(indexMAX));
% end

%% Using FFT to calculate Bode Plot values
MagFFT = zeros(1,40); %Preallocated for speed
PhaseFFT = zeros(1,40); %Preallocated for speed
for i = 1 :40
    rex = (Dataset(i).Y(1).Data); %Rexroth Output
    cmd = (Dataset(i).Y(2).Data); %Commanded

    N = length(cmd);
    Xcmd = fft(cmd)/N;
    Xrex = fft(rex)/N;

    XcmdT = Xcmd(1:round(N/2)); %Truncated FFT
    XrexT = Xrex(1:round(N/2)); %Truncated FFT

    [XcmdMax,indexXcmdMax] = max(abs(XcmdT));
    [XrexMax,indexXrexMax] = max(abs(XrexT));

    %check for the same
    if (indexXcmdMax ~= indexXrexMax)
        error('Something screwy, frequency mismatch of maximum points ',...
              'from FFT analysis of LTB position input', ...
              'to Rexroth position output waveforms.');
    end

    MagFFT(i) = 20*log(abs(Xrex(indexXrexMax))...
                      /abs(Xcmd(indexXcmdMax)));
    PhaseFFT(i) = -(angle(Xcmd(indexXcmdMax))-
                        angle(Xrex(indexXrexMax)))*180/pi;
end

%If Phase is leading, converts to lagging
%(corrects artifact of incorrect phase angles)
if PhaseFFT(i) > 180
    PhaseFFT(i) = PhaseFFT(i) - 360;
end

%%%Plot the FFT
if (0)
    figure(4);
    subplot(2,1,1)
    plot([0:N-1],abs(Xcmd),[0:N-1],abs(Xrex))
    legend('Cmd','Rex');
    title(['for i ',num2str(i)])
    subplot(2,1,2)
    plot([0:N-1],angle(Xcmd)*180/pi,[0:N-1],angle(Xrex)*180/pi)
end
end

%% Mean-Squared Error
% Calculated between FFT-based Transfer Function and
% 1st Order Estimated Transfer Function
for wc = 0.01:0.01:2
    s=tf('s');
    H = 1/(s/wc+1);
    [MAG,PHASE] = bode(H,frequencies');
    MAG = squeeze(MAG);
    MagTxfrSwp = 20*log10(MAG)';
    PhaseTxfrSwp = squeeze(PHASE)';
    index = cast(wc*100,'uint8');
    MSE.mag((index)) = mean((MagTxfrSwp - MagFFT).^2);
    MSE.phase((index)) = mean((PhaseTxfrSwp - PhaseFFT).^2);
end

[MSE.magMINvalue,MSE.magMINindex] = min(MSE.mag); % Mean-Squared Error Magnitude
[MSE.phaseMINvalue,MSE.phaseMINindex] = min(MSE.phase);

MSE_fig = figure;
subplot(2,1,1),plot([0.01:0.01:2],MSE.mag);
title(['MSE Magnitude, Minimum Value at \omega_c = ', num2str(MSE.magMINindex/100)]);
xlabel('\omega_c value');
ylabel('Mean Squared Error');

subplot(2,1,2),plot([0.01:0.01:2],MSE.phase);
title(['MSE Phase, Minimum Value at \omega_c = ', num2str(MSE.phaseMINindex/100)]);
xlabel('\omega_c value');
ylabel('Mean Squared Error');

%% Official Transfer function Estimate of LTB
%First order transfer function magnitude (estimate of LTB p-to-p

% s=tf('s');
% wc = 0.8;
% H = 1/(s/wc+1);
% [MAG,PHASE] = bode(H,frequencies');
% MAG = squeeze(MAG);
% MagTRANSFERest = 20*log10(MAG)';
% PhaseTRANSFERest(1) = squeeze(PHASE)';

s=tf('s');
wc = 0.4;
H = 1/(s/wc+1);
[MAG.p4,PHASE.p4] = bode(H,frequencies');
MAG.p4 = squeeze(MAG.p4);
MagTRANSFERest.p4 = 20*log10(MAG.p4)';
PhaseTRANSFERest.p4 = squeeze(PHASE.p4)';

wc = 0.5;
H = 1/(s/wc+1);
[MAG.p5,PHASE.p5] = bode(H,frequencies');
MAG.p5 = squeeze(MAG.p5);
MagTRANSFERest.p5 = 20*log10(MAG.p5)';
PhaseTRANSFERest.p5 = squeeze(PHASE.p5)';

wc = 0.6;
H = 1/(s/wc+1);
[MAG.p6,PHASE.p6] = bode(H,frequencies');
MAG.p6 = squeeze(MAG.p6);
MagTRANSFERest.p6 = 20*log10(MAG.p6)';
PhaseTRANSFERest.p6 = squeeze(PHASE.p6)';

wc = 0.7;
H = 1/(s/wc+1);
[MAG.p7,PHASE.p7] = bode(H,frequencies');
MAG.p7 = squeeze(MAG.p7);
MagTRANSFERest.p7 = 20*log10(MAG.p7)';
PhaseTRANSFERest.p7 = squeeze(PHASE.p7)';

wc = 0.8;
H = 1/(s/wc+1);
[MAG.p8,PHASE.p8] = bode(H,frequencies');
MAG.p8 = squeeze(MAG.p8);
MagTRANSFERest.p8 = 20*log10(MAG.p8)';
PhaseTRANSFERest.p8 = squeeze(PHASE.p8)';

%% Bode Plot Construction
if (0)
    figure;
    hold on;
    subplot(2,1,1), semilogx(frequencies,MagTRANSFERest.p4,'--ob', ... 
                       frequencies,MagTRANSFERest.p5,'--og', ... 
                       frequencies,MagTRANSFERest.p6,'--oc', ... 
                       frequencies,MagTRANSFERest.p7,'--oy', ... 
                       frequencies,MagTRANSFERest.p8,'--om', ... 
                       frequencies,MagFFT, '-+r');
    legend( 'H = 1/(s/0.4+1)', ... 
           'H = 1/(s/0.5+1)', ... 
           'H = 1/(s/0.6+1)', ... 
           'H = 1/(s/0.7+1)', ... 
           'H = 1/(s/0.8+1)', ... 
           'Bode from FFT');
    ylabel('Gain in [dB]');
    xlabel('Frequency [Hz]');
    title('Gain Plot for LTB w/ L-10 Position (dSPACE-in, Rexroth-out)');
    grid on;
    axis([0.1 4 -40 2]);

    subplot(2,1,2), semilogx(frequencies,PhaseTRANSFERest.p4,'--ob', ... 
                           frequencies,PhaseTRANSFERest.p5,'--og', ... 
                           frequencies,PhaseTRANSFERest.p6,'--oc', ... 
                           frequencies,PhaseTRANSFERest.p7,'--oy', ... 
                           frequencies,PhaseTRANSFERest.p8,'--om', ... 
                           frequencies,PhaseFFT, '-+r');
    legend( 'H = 1/(s/0.4+1)', ... 
            'H = 1/(s/0.5+1)', ... 
            'H = 1/(s/0.6+1)', ... 
            'H = 1/(s/0.7+1)', ... 
            'H = 1/(s/0.8+1)', ... 
            'Bode from FFT');
    ylabel('Angle [degrees]');
    xlabel('Frequency [Hz]');
    title('Phase Plot for LTB w/ L-10 Position (dSPACE-in, Rexroth-out)');
    grid on;
    axis([0.1 4 -120 0]);
```matlab
figure;
hold on;
subplot(2,1,1), semilogx(frequencies, MagSINEpos, '-og', ...
    frequencies, MagSINEneg, ':ob', ...
    frequencies, MagTRANSFERest.p8, '--or', ...
    frequencies, MagFFT, '-.c');
legend( 'Positive Sine Amplitude', ...
    'Negative Sine Amplitude', ...
    '1st order Estimate T(s)', ...
    'Bode from FFT');
ylabel('Gain in [dB]');
xlabel('Frequency [Hz]');
title('Gain Plot for LTB w/ L-10 Position 
(dSPACE-in, Rexroth-out)');
grid on;
axis([0.1 4 -40 2]);

subplot(2,1,2), semilogx(frequencies, PhaseSINEpos, '-og', ...
    frequencies, PhaseSINEneg, ':ob', ...
    frequencies, PhaseTRANSFERest.p8, '--or', ...
    frequencies, PhaseFFT, '-.c');
legend( '@ Sine = 0, Positive Slopes', ...
    '@ Sine = 0, Negative Slopes', ...
    '1st order Estimate T(s)', ...
    'Bode from FFT');
ylabel('Angle [degrees]');
xlabel('Frequency [Hz]');
title('Phase Plot for LTB w/ L-10 Position 
(dSPACE-in, Rexroth-out)');
grid on;
axis([0.1 4 -120 0]);
end

%% Plotting Time-Domain and Frequency-Domain Bode Plots
if (0)
    figure;
    hold on;
    subplot(2,1,1), semilogx(frequencies, MagSINEpos, '-g', ...
        frequencies, MagSINEneg, '-.b', ...
        frequencies, MagFFT, '--r');
    legend('from Positive Sine Peaks', ...
        'from Negative Sine Peaks', ...
        'from FFT Analysis');
ylabel('Gain in [dB]');
xlabel('Frequency [Hz]');
```
title('Gain Plot for LTB w/ L-10 Position (dSPACE-in, Rexroth-out)');
grid on;
axis([0.1 4 -40 2]);

subplot(2,1,2), semilogx(frequencies, PhaseSINEpos, '-g', ...
frequencies, PhaseSINEneg, '-.b', ...
frequencies,PhaseFFT, '--r');
legend('from Zero Crossing Positive Slopes', 'from Zero Crossing Negative Slopes', ...
'from FFT Analysis');
ylabel('Angle [degrees]');
xlabel('Frequency [Hz]');
title('Phase Plot for LTB w/ L-10 Position (dSPACE-in, Rexroth-out)');
grid on;
axis([0.1 4 -120 0]);

end

%% Plotting just FFT
if (1)
    figure;
    hold on;
    subplot(2,1,1), semilogx(frequencies,MagFFT, '-or');
    legend('from FFT Analysis');
ylabel('Gain in [dB]');
xlabel('Frequency [Hz]');
title('Gain Plot for LTB w/ L-10 Position (dSPACE-in, Rexroth-out)');
grid on;
axis([0.1 4 -40 2]);

    subplot(2,1,2), semilogx(frequencies,PhaseFFT, '-or');
    legend('from FFT Analysis');
ylabel('Angle [degrees]');
xlabel('Frequency [Hz]');
title('Phase Plot for LTB w/ L-10 Position (dSPACE-in, Rexroth-out)');
grid on;
axis([0.1 4 -120 0]);
end

%% Plotting just FFT
.4 Force Transfer Function Characterization

Table 1: Cable Pin-Out for Control Computer to Load Cell Signal Amplifier

<table>
<thead>
<tr>
<th>Signal</th>
<th>Pin</th>
<th>Source Connector</th>
<th>Pin</th>
<th>Destination Connector</th>
</tr>
</thead>
<tbody>
<tr>
<td>Output (+)</td>
<td>Center</td>
<td>#1 BNC (Male)</td>
<td>B</td>
<td>PC06W-10-6S(SR)</td>
</tr>
<tr>
<td>Output (-)</td>
<td>Center</td>
<td>#2 BNC (Male)</td>
<td>C</td>
<td>PC06W-10-6S(SR)</td>
</tr>
<tr>
<td>Ground</td>
<td>Case</td>
<td>#1 BNC (Male)</td>
<td>D</td>
<td>PC06W-10-6S(SR)</td>
</tr>
<tr>
<td>Ground</td>
<td>Case</td>
<td>#1 BNC (Male)</td>
<td>D</td>
<td>PC06W-10-6S(SR)</td>
</tr>
</tbody>
</table>

.4.1 Simulink Model

Figure 7: Force Transfer Function Simulink Model
.4.2 ControlDesk Interface

![ControlDesk Interface](image)

Figure 8: Force Transfer Function ControlDesk Interface

.4.3 MATLAB Code for Initialization of Simulink Blocks

```matlab
% Initialization file
% for LTB Transfer Function Creation
% of Load Cell Signal moving through System
%(from output of load cell ((MIL connector)))
% to
%(BNC Left Load Cell & Right Load Cell) on side
%of control cabinet
%Creation: Chris Haller, 2010 April 5

% 5000 LBS Load-Cells

LeftCell_gain = 5000*4.44822*1.00130;
LeftCell_zero_offset = 5+9.9;
```
137

```matlab
RightCell_gain = 5000*4.44822*0.996359;
RightCell_zero_offset = 13.5+6.5;

 RightCell
% 1000 LBS Load-Cells
%LeftCell_gain = 5000*4.44822*1.00130;;
%LeftCell_zero_offset = 5;
%RightCell_gain = 5000*4.44822*0.996359;
%RightCell_zero_offset = 13.5;

 % 250 LBS Load-Cells
%LeftCell_gain = 250*4.44822*1.00130;
%LeftCell_zero_offset = 6.5;
%RightCell_gain = 250*4.44822*1.00450;
%RightCell_zero_offset = 6.5;

disp('transfer_function_init.m complete')
```

.4.4 MATLAB Code for Calculation of Force Transfer Function

```matlab
%====================================================================
% Load Cell (force) Transfer Function Calculator
% Christopher Haller
% Created: 2010 April 7

% Usage:
% force_transfer_function.m
% Determines transfer function for the
% linear test bed load-cell cuitry

% Transfer Function Input is the load-cell
% connection point on amplifier
% box located on top of LTB carriage Arm
% (DMA Inputs from L.C.s (Out+) and (Out-))

% Transfer Function Output is the left (and right) load cell BNC
% connection points on the sides of the LTB control cabinet.

% Signal evaluation performed with dSPACE.
```
% LLC Signal Rating: 2.048 mV/V
% RLC Signal Rating: 2.041 mV/V
% Original Excitation Voltage: 10 VDC
% Voltage level for neutral point of differential voltage: 5 VDC

%% Initialization of MATLAB
close all;
clear;
c1c;
set(0, 'DefaultTextFontSize', 16) ;
set(0, 'DefaultAxesFontSize', 16) ;
set(0, 'DefaultLineLineWidth', 2) ;
set(0, 'DefaultAxesXGrid', 'on') ;
set(0, 'DefaultAxesYGrid', 'on') ;

%% Program Operation Choices.
% Dataset to Load (choose 1)
a=1; % llc_neg2048_10sec.mat
b=0; % llc_pos2048_10sec.mat
c=0; % llc_pos2_10sec.mat
d=0; % llc_neg2_10sec.mat
e=0; % rlc_neg2041_10sec.mat
f=0; % rlc_pos2041_10sec.mat
g=0; % rlc_pos2_10sec.mat
h=0; % rlc_neg2_10sec.mat

% Show sweep of damping coefficient
visual_sweep = 1;

% Program Mode
automatic_damp = 0;
manual_damp = 1;

% Load Desired Data into Dataset Struct variable
if a==1
    S = load('MATLAB_Data\llc_neg2048_10sec.mat');
end
if b==1
    S = load('MATLAB_Data\llc_pos2048_10sec.mat');
end
if c==1
    S = load('MATLAB_Data\llc_pos2_10sec.mat');
end
if d==1
    S = load('MATLAB_Data\llc_pos2_10sec.mat');
end
if e==1
    S = load('MATLAB_Data\llc_neg2_10sec.mat');
end
if f==1
    S = load('MATLAB_Data\rlc_neg_2041_10sec.mat');
end
if g==1
    S = load('MATLAB_Data\rlc_pos_2041_10sec.mat');
end
if h==1
    S = load('MATLAB_Data\rlc_pos_2_10sec.mat');
end
if d==1
S = load('MATLAB_Data\llc_neg2_10sec.mat');
end
if e==1
S = load('MATLAB_Data\rlc_neg2041_10sec.mat');
end
if f==1
S = load('MATLAB_Data\rlc_pos2041_10sec.mat');
end
if g==1
S = load('MATLAB_Data\rlc_pos2_10sec.mat');
end
if h==1
S = load('MATLAB_Data\rlc_neg2_10sec.mat');
end

names = fieldnames(S);

%Dataset.Y(1) = Left Load Cell
%Dataset.Y(2) = Right Load Cell
%Dataset.Y(3) = Step Output (Control Step)
Dataset = S.(names{1,1});

%% Plot all Time Domain Data for given input file
if 1

%Left Load Cell & Control Step
if a || b || c || d
figure;
subplot(2,1,1), plot(Dataset.X.Data,Dataset.Y(1).Data);
xlabel('time (seconds)');
ylabel('LLC Output (volts)');
title('LLC Time Domain Data');
subplot(2,1,2), plot(Dataset.X.Data,Dataset.Y(3).Data);
xlabel('time (seconds)');
ylabel('Step Input');
end

%Right Load Cell & Control Step
if e || f || g || h
figure;
subplot(2,1,1), plot(Dataset.X.Data,Dataset.Y(2).Data);
xlabel('time (seconds)');
ylabel('RLC Output (volts)');
title('RLC Time Domain Data');
subplot(2,1,2), plot(Dataset.X.Data,Dataset.Y(3).Data);
xlabel('time (seconds)');
ylabel('Step Input');

end

end

%% Manual Damping Chosen, Load Cell Transfer Function

if manual_damp

% Left Load Cell & Control Step
if a || b || c || d

damping = 0.32; % 0.29; % Left Load Cell Damping Ratio
end

% Right Load Cell & Control Step
if e || f || g || h

damping = 0.32; % 0.36; % Right Load Cell Damping Ratio
end

s = tf('s');
w = 3.927e3;
H = 1/(s^2/w^2 + 2*damping*s/w + 1)*exp(-s*100e-6);
% Extract row vector formatted num & den from H
[num, den] = tfdata(H, 'v');
% Transfer Function to State Space Conversion
[A, B, C, D] = tf2ss(num, den);
% Simulate time response of LTI model to arbitrary input (Dataset.Y(3))
Y = lsim(H, Dataset.Y(3).Data/-0.02, Dataset.X.Data);

end

%% Automatic Damping Ratio Sweep for Load Cell
%% Transfer Function Matching

if automatic_damp

figure;

for damping = 0.01 : 0.01 : 1

index = cast(damping*100, 'uint8');

s = tf('s');
w = 3.927e3;
H = 1/(s^2/w^2 + 2*damping*s/w + 1)*exp(-s*100e-6);
% Extract row vector formatted num & den from H
[num, den] = tfdata(H, 'v');
% Transfer Function to State Space Conversion
[A, B, C, D] = tf2ss(num, den);
% Simulate time response of LTI model to arbitrary input (Dataset.Y(3))
Y = lsim(H,Dataset.Y(3).Data/-0.02,Dataset.X.Data);

if a || b || c || d %Left Cell
    %Mean Squared Error
    maxLLCforce = abs(max(Dataset.Y(1).Data));
    if b || c %for pos vs. neg waveform
        maxLLCforce = abs(min(Dataset.Y(1).Data));
    end

%Normalize Force
    LLCnormal = Dataset.Y(1).Data'./maxLLCforce;
    maxY=abs(min(Y));
    Ynormal = Y./maxY; %Normalize Y Data

%find LLC min peak
    [minLLCval,minLLCindex] = min(Dataset.Y(1).Data);
    if b || c %for pos vs. neg waveform
        %find LLC min peak
        [minLLCval,minLLCindex] = max(Dataset.Y(1).Data);
    end

    MSE(index) = mean((LLCnormal((minLLCindex+0):...)
    (minLLCindex+15)) - Y((minLLCindex+0): ...
    (minLLCindex+15))).^2); %eval step in range

    if visual_sweep
        plot(Dataset.X.Data,LLCnormal,...
        Dataset.X.Data,Y);
        axis([Dataset.X.Data(minLLCindex-15) ...
        Dataset.X.Data(minLLCindex+75) ... -1 1]);
        title(['Sweeping through \zeta values of underdamped system :
', num2str(damping)]);
        xlabel('time (seconds)');
        ylabel('normalized input, output, and estimation');
        pause(0.000001);
    end
end

if e || f || g || h %Right Load Cell

    %Mean Squared Error
    minRLCforce = abs(max(Dataset.Y(2).Data));
    if f || g %for pos
        minRLCforce = abs(min(Dataset.Y(2).Data));
RLCnormal = Dataset.Y(2).Data'./minRLCforce; %Normalize Force
maxY=abs(min(Y));
Ynormal = Y./maxY; %Normalize Y Data

%find LLC min peak
[minRLCval,minRLCindex] = min(Dataset.Y(2).Data);
if f || g %for pos
    [minRLCval,minRLCindex] = max(Dataset.Y(2).Data);
end

MSE(index) = mean((RLCnormal((minRLCindex+0): ...
    (minRLCindex+15)) - Y((minRLCindex+0): ...
    (minRLCindex+15))).^2); %eval step in range
if visual_sweep
    plot(Dataset.X.Data,RLCnormal,...
        Dataset.X.Data,Y);
    axis([Dataset.X.Data(minRLCindex-15) ...
        Dataset.X.Data(minRLCindex+75) ...
        -1 1]);
    title(['Sweeping through \zeta values of underdamped 
        system : ', num2str(damping)]);
    xlabel('time (seconds)');
    ylabel('normalized input, output, and estimation');
    pause(0.000001);
end

figure;
plot(MSE);
title('Mean Squared Error');
xlabel('Index # corresponding to \zeta');
ylabel('MSE value');

% Create Load Cell Transfer Function based on best DAMPING RATIO
[MSEval,MSEminINDEX] = min(MSE);
damping = MSEminINDEX/100;
s=tf('s');
w = 3.927e3;
H = 1/(s^2/w^2 + 2* damping*s/w + 1) * exp(-s*100e-6);
%Extract row vector formatted num & den from H
[num, den] = tfdata(H, 'v');
% Transfer Function to State Space Conversion
[A, B, C, D] = tf2ss(num, den)
SSforce = ss(A, B, C, D);
SSforce.InputDelay = 100e-6;

% Adding time delay to TF (for plotting purposes)
H = 1/(s^2/w^2 + 2*damping*s/w + 1)*exp(-s*100e-6);

% Simulate time response of LTI model
% to arbitrary input (Dataset.Y(3))
Y = lsim(H, Dataset.Y(3).Data/-0.02, Dataset.X.Data);
end

%% Plot Transfer Function & actual results
if 1
    % Left Load Cell & Control Step
    if a || b || c || d
        final_figure = figure;
        plot(Dataset.X.Data, Dataset.Y(1).Data/15000, '-r', ...
             Dataset.X.Data, Dataset.Y(3).Data/-0.02, '-g', ...
             Dataset.X.Data, Y, '-.b');
        legend('Left Load Cell Output', 'Step Input', 'Transfer Function');
        [minLLCval, minLLCindex] = min(Dataset.Y(1).Data);
        [maxLLCval, maxLLCindex] = max(Dataset.Y(1).Data);
        axis([Dataset.X.Data(minLLCindex-15) ...
             Dataset.X.Data(minLLCindex+50) ...
             -inf inf]);
        if b || c
            axis([Dataset.X.Data(maxLLCindex-15) ...
                  Dataset.X.Data(maxLLCindex+50) ...
                  -inf inf]);
        end
        title(['LLC, 2nd order system, with \omega_N = ', num2str(w), '
               and \zeta = ', num2str(damping)]);
        xlabel('time (seconds)');
        ylabel('normalized amplitude');
    end

    % Right Load Cell & Control Step
    if e || f || g || h
        final_figure = figure;
        plot(Dataset.X.Data, Dataset.Y(2).Data/15000, '-r', ...
             Dataset.X.Data, Dataset.Y(3).Data/-0.02, '-g', ...
Dataset.X.Data,Y,'-.b');
legend('Right Load Cell Output','Step Input','Transfer Function');
[minRLCval,minRLCindex] = min(Dataset.Y(2).Data);
[maxRLCval,maxRLCindex] = max(Dataset.Y(2).Data);
axis([Dataset.X.Data(minRLCindex-15) ...
     Dataset.X.Data(minRLCindex+50) ... 
    -inf inf]);
if f || g
    axis([Dataset.X.Data(maxRLCindex-15) ...
          Dataset.X.Data(maxRLCindex+50) ...
          -inf inf]);
end
title(['RLC, 2nd order system, with \omega_N =', num2str(w),'
and \zeta = ',num2str(damping)]);
xlabel('time (seconds)');
ylabel('normalized amplitude');
end
end

%% Plotting just step response.
if 0
    figure;
    %Left Load Cell & Control Step
    if a || b || c || d
        plot(Dataset.X.Data,Dataset.Y(1).Data/15000,'-.r',...
             Dataset.X.Data,Dataset.Y(3).Data/-0.02,'-b');
        legend('Left Load Cell Output','Step Input');
        [minLLCval,minLLCindex] = min(Dataset.Y(1).Data);
        [maxLLCval,maxLLCindex] = max(Dataset.Y(1).Data);
        axis([Dataset.X.Data(minLLCindex-15) ...
              Dataset.X.Data(minLLCindex+50) ...
             -inf inf]);
        if b || c
            axis([Dataset.X.Data(maxLLCindex-15) ...
                  Dataset.X.Data(maxLLCindex+50) ...
                  -inf inf]);
        end
        title('Normalized Step Response of Left Load Cell');
        xlabel('time (seconds)');
        ylabel('normalized amplitude');
    end
    %Right Load Cell & Control Step
    if e || f || g || h
        plot(Dataset.X.Data,Dataset.Y(2).Data/15000,'-.r',...

Dataset.X.Data,Dataset.Y(3).Data/-0.02,'-b');
legend('Right Load Cell Output','Step Input');
[minRLCval,minRLCindex] = min(Dataset.Y(2).Data);
[maxRLCval,maxRLCindex] = max(Dataset.Y(2).Data);
axis([Dataset.X.Data(minRLCindex-15) ...
      Dataset.X.Data(minRLCindex+50) ...]
     -inf inf]);
if f || g
  axis([Dataset.X.Data(maxRLCindex-15) ...
       Dataset.X.Data(maxRLCindex+50) ...]
       -inf inf]);
end
title('Normalized Step Response of Right Load Cell');
xlabel('time (seconds)');
ylabel('normalized amplitude');
end
end
.5 Noise Covariance Characterization

.5.1 Simulink Model

Figure 9: Noise Characterization Simulink Model
.5.2 ControlDesk Interface

![ControlDesk Interface](image)

Figure 10: Noise Characterization ControlDesk Interface

.5.3 MATLAB Code for Calculation of Noise Covariance

```matlab
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% OSU LINEAR TEST BED (LTB) Variance Calculations
% Created: Christopher Haller, 2010 March 5
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Note: Signal measurements based upon linear test bed energized with
% carriage homed and in centered stationary (non-moving) position.
% Measurements taken with dSPACE 1103

%% Initialization
clear
clc
close all;
```
format compact

%% Variables:
% V1 = Variance from Dataset #1 by MATLAB's "var"
% Me1 = Variance from Dataset #1 by Chris
% .p = Position Signal Variance
% .v = Velocity Signal Variance
% .a = Acceleration Signal Variance
% .llc = Left Load Cell Signal Variance
% .rlc = Right Load Cell Signal Variance

set(0, 'DefaultTextFontSize', 16) ;
set(0, 'DefaultAxesFontSize', 16) ;
set(0, 'DefaultLineLineWidth', 2) ;
set(0, 'DefaultAxesXGrid', 'on') ;
set(0, 'DefaultAxesYGrid', 'on') ;

%% Measured or assumed

k.base.time = 1;
% 1.3 [M-N/m] spring force (motor cog, yoke flex)
k.base.spring = 1.3e6;
k.base.length = 1; %[m] stroke of LTB
k.base.force = 10e3; % [N]

%Dependent bases
k.base.velocity = k.base.length / k.base.time;
k.base.acceleration = k.base.velocity / k.base.time;

%Other Variables
% Left load cell (5000 lb)
LeftCell_geometry_scaling = 2/3;
LeftCell_gain = 5000*4.44822*1.00130;
LeftCell_zero_offset = 5+9.9;
LeftArm_mass_offset = 7.125;

% Right load cell (5000 lb)
RightCell_geometry_scaling = 2/3;
RightCell_gain = 5000*4.44822*0.996359;
RightCell_zero_offset = 13.5+6.5;
RightArm_mass_offset = 7.125; % (kg) Use to calibrate zero

% Accel
Acceleration_gain = 5.00;
Acceleration_offset = 0;

% Speed
Speed_gain = 3.0003; %(1000 and 5000 lb)
% Speed_gain = 3.000; %(250 lb)
Speed_offset = +.001; %(500 lb)
% Speed_offset = 0; %(250 and 1000 lb)
% Position
Position_gain = 1.00290;
Position_offset = 0; %(1000 and 5000 lb)
% Position_offset = -0.0007; %(250 lb)
Yoke_mass_offset = 35.25; %(1000 and 5000 lb)
% Yoke_mass_offset = 0; %(250 lb)
DUT_mass = 1272.5; % (kg)
% DUT_mass = 0; %(250 lb)
Gravity_accel = 9.80665;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Covariance calculations
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

ecycle Choose one of the following 5 static noise measurement files to load:

cycle % Choice #1
load('static_noise_pvalr_1000persec_1200sec');
n = static_noise_pvalr_1000persec_1;
cycle % Choice #2
load('static_noise_pvalr_10000persec_1sec');
n = static_noise_pvalr_10000persec_;
cycle % Choice #3
load('static_noise_pvalr_10000persec_10sec');
n = static_noise_pvalr_10000persec_;
cycle % Choice #4
load('static_noise_pvalr_10000persec_60sec');
n = static_noise_pvalr_10000persec_;
cycle % Choice #5
load('static_noise_pvalr_10000persec_200sec');
n = static_noise_pvalr_10000persec_;

cycle Time domain noise data extraction in column format (Transposed ')
for i = 1:5
if strcmp(noise.Y(i).Name, '"Labels"/"raw_ADC_Acceleration"')
Na = noise.Y(i).Data';
end %elif
else if strcmp(noise.Y(i).Name, '"Labels"/"raw_ADC_Velocity"')
Nv = noise.Y(i).Data';
end %elif
103     elseif strcmp(noise.Y(i).Name, "Labels"/"raw_ADC_Position")
104         Np = noise.Y(i).Data';
105     elseif strcmp(noise.Y(i).Name, "Labels"/"raw_ADC_Right LC")
106         Nl = noise.Y(i).Data';
107     elseif strcmp(noise.Y(i).Name, "Labels"/"raw_ADC_Left LC")
108         Nr = noise.Y(i).Data';
109     end %if
110 end

111 % Nt = [Np Nv Na Nc]; %Time-Domain Noise Matrix
112 %position = 1
113 %velocity = 2
114 %acceleration = 3
115 %force = 4
116 Ntpu(:,1) = (Np*Position_gain + Position_offset)/k.base.length;
117 Ntpu(:,2) = (Nv*Speed_gain + Speed_offset)/k.base.velocity;
118 Ntpu(:,3) = (Na*Acceleration_gain + ... Acceleration_offset)/k.base.acceleration;
119 Ntpu(:,4) = (Nr*RightCell_geometry_scaling*RightCell_gain + ... LeftCell_zero_offset + Nl*LeftCell_geometry_scaling*... LeftCell_gain + RightCell_zero_offset)/k.base.force;

120 %Covariance of Measurement Noise /?/ Variance of Measurement Noise
121 %%% pos vel acc loadcell
122 Rn = [covar(Ntpu(:,1),Ntpu(:,1)) covar(Ntpu(:,1),Ntpu(:,2)) ... covar(Ntpu(:,1),Ntpu(:,3)) covar(Ntpu(:,1),Ntpu(:,4)) ; ...
123      covar(Ntpu(:,2),Ntpu(:,1)) covar(Ntpu(:,2),Ntpu(:,2)) ... covar(Ntpu(:,2),Ntpu(:,3)) covar(Ntpu(:,2),Ntpu(:,4)) ; ...
124      covar(Ntpu(:,3),Ntpu(:,1)) covar(Ntpu(:,3),Ntpu(:,2)) ... covar(Ntpu(:,3),Ntpu(:,3)) covar(Ntpu(:,3),Ntpu(:,4)) ; ...
125      covar(Ntpu(:,4),Ntpu(:,1)) covar(Ntpu(:,4),Ntpu(:,2)) ... covar(Ntpu(:,4),Ntpu(:,3)) covar(Ntpu(:,4),Ntpu(:,4))];

126 %Rn= cov(Ntpu);  %%%MATLAB's way of calculating covariance, minor
127 %differences in results from Chris' method of calculating variance.
128
129 % Plotting Section
130 %LTB Gaussian Noise on Feedback Signal Lines
131 if 0
132    subplot(2,2,1), plot(noise.X.Data,Np);
MATLAB Code for Calculation of Kalman Filter

```matlab
function [c] = covar(X,Y)

% Created by CAH, 2010 April 15
% Calculates covariance of X and Y datasets
% covariance(X,Y) = sum ((x_i-x_avg)*(y_i-y_avg))/N
Nx = length(X);
Ny = length(Y);
if Nx ~= Ny
    error(' Data Length Mismatch for covariance calculation. ');
end
c = (sum((X-avg(X)).*(Y-avg(Y))))/Nx;
end

.6 MATLAB Code for Calculation of Kalman Filter
```
% Other Variables
% Left load cell (5000 lb)
LeftCell_geometry_scaling = 2/3;
LeftCell_gain = 5000*4.44822*1.00130;
LeftCell_zero_offset = 5+9.9;
LeftArm_mass_offset = 7.125;
% Right load cell (5000 lb)
RightCell_geometry_scaling = 2/3;
RightCell_gain = 5000*4.44822*0.996359;
RightCell_zero_offset = 13.5+6.5;
RightArm_mass_offset = 7.125; % (kg) Use to calibrate zero
% Accel
Acceleration_gain = 5.00;
Acceleration_offset = 0;
% Speed
Speed_gain = 3.0003; % (1000 and 5000 lb)
% Speed_gain = 3.000; % (250 lb)
Speed_offset = +.001; % (5000 lb)
% Speed_offset = 0; % (250 and 1000 lb)
% Position
Position_gain = 1.00290;
Position_offset = 0; % (1000 and 5000 lb)
% Position_offset = -0.0007; % (250 lb)
Yoke_mass_offset = 35.25; % (1000 and 5000 lb)
% Yoke_mass_offset = 0; % (250 lb)
DUT_mass = 1272.5; % (kg)
% DUT_mass = 0; % (250 lb)
Gravity_accel = 9.80665;

%% System equations:
% Zm = Hp * Zc
% Fltbm = Hf * Fltb + V
% Fltb = (Zm - Zb) * Kltb
% Transfer functions:
% Hp = e^(-s*Tdp) * (Wp / (s + Wp))
% Hf = e^(-s*Tdf) * (1/(s^2/Wf^2 + 2*damp*s/Wf + 1))
% First-Order Approx. (w/o time-delay or random measurement noise):
% sFltb = W (stochastic variable)
% System equations (simplified Zmm = Zm) aka Hm = 1:
% Zm = Hp * Zc
% Fltbm = Hf * Fltb + V
\[ F_{tb} = (Z_m - Z_b) \cdot K_{ltb} \]
\[ V_m = s \cdot Z_m / (s/W_v + 1) \]
\[ s \cdot Z_m = V_m \cdot (s/W_v + 1) \]
\[ s \cdot W_v \cdot Z_m = V_m \cdot (s + W_v) \]
\[ \Rightarrow s \cdot V_m = s \cdot W_v \cdot Z_m - V_m \cdot W_v \]
\[ A_m = s \cdot V_m / (s/W_a + 1) \]
\[ A_m(s/W_a + 1) = s \cdot V_m \]
\[ A_m \cdot s + A_m \cdot W_a = s \cdot V_m \cdot W_a \]
\[ \Rightarrow s \cdot A_m = s \cdot V_m \cdot W_a - A_m \cdot W_a \]

%% System modeling with process disturbances and measurement noise
\[ \dot{x} = Ax + Bu + Gw \quad \text{\{State equation\}} \]
\[ y = Cx + Du + Hw + v \quad \text{\{Measurements\}} \]

KALMAN takes the state-space model SYS=SS(A,[B G],C,[D H]) and the covariance matrices:
\[ Q_n = E\{ww'\}, \quad R_n = E\{vv'\}, \quad N_n = E\{wv'\} \]
\[ \{KEST,L,P\} = \text{kalman}(SYS,Q_n,R_n,N_n) \]


\%% Covariance calculations

\% Choose one of the following 5 sfiles to load:
\% load('Noise\Measurements\static\noise\pvalr\1000persec\1200sec');
\% load('Noise\Measurements\static\noise\pvalr\10000persec\1sec');
\% load('Noise\Measurements\static\noise\pvalr\10000persec\10sec');
\% load('Noise\Measurements\static\noise\pvalr\10000persec\60sec');
\% load('Noise\Measurements\static\noise\pvalr\10000persec\200sec');
\%names = fieldnames(NoiseData);
\%noise = NoiseData.(names{1,1});
\%noise = static\noise\pvalr\10000persec.;
\%Time domain noise data extraction in column format (Transposed )
\for i = 1:5
\if strcmp(noise.Y(i).Name, '"Labels"/"raw_APC_Acceleration"')
\ Na = noise.Y(i).Data';
\elseif strcmp(noise.Y(i).Name, '"Labels"/"raw_APC_Velocity"')
\ Nv = noise.Y(i).Data';
\elseif strcmp(noise.Y(i).Name, '"Labels"/"raw_APC_Position"')
\ Np = noise.Y(i).Data';
\elseif strcmp(noise.Y(i).Name, '"Labels"/"raw_APC_Right_LC"')
\ Nl = noise.Y(i).Data';
\elseif strcmp(noise.Y(i).Name, '"Labels"/"raw_APC_Left_LC"')
\ Nr = noise.Y(i).Data';
\fi
end if
end

% Nt = [Np Nv Na Nc]; %Time-Domain Noise Matrix
%position = 1
%velocity = 2
%acceleration = 3
%force = 4
Ntpu(:,1) = (Np*Position\_gain + Position\_offset)/k.base.length;
Ntpu(:,2) = (Nv*Speed\_gain + Speed\_offset)/k.base.velocity;
Ntpu(:,3) = (Na*Acceleration\_gain + ...
Accelerate\_offset)/k.base.acceleration;
Ntpu(:,4) = (Nr*RightCell\_geometry\_scaling*RightCell\_gain + ...
LeftCell\_zero\_offset + Nl*LeftCell\_geometry\_scaling*...
LeftCell\_gain + RightCell\_zero\_offset)/k.base.force;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% pos vel acc loadcell
Rn = [covar(Ntpu(:,1),Ntpu(:,1)) covar(Ntpu(:,1),Ntpu(:,2)) ...
covar(Ntpu(:,1),Ntpu(:,3)) covar(Ntpu(:,1),Ntpu(:,4)) ;
... 
covar(Ntpu(:,2),Ntpu(:,1)) covar(Ntpu(:,2),Ntpu(:,2)) ...
covar(Ntpu(:,2),Ntpu(:,3)) covar(Ntpu(:,2),Ntpu(:,4)) ;
... 
covar(Ntpu(:,3),Ntpu(:,1)) covar(Ntpu(:,3),Ntpu(:,2)) ...
covar(Ntpu(:,3),Ntpu(:,3)) covar(Ntpu(:,3),Ntpu(:,4)) ;
... 
covar(Ntpu(:,4),Ntpu(:,1)) covar(Ntpu(:,4),Ntpu(:,2)) ...
covar(Ntpu(:,4),Ntpu(:,3)) covar(Ntpu(:,4),Ntpu(:,4))];

%%%%%%%%%%%%%%%%%%%%%%MATLAB's way of calculating covariance, minor
% differences in results from Chris' method of calculating variance.

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% State Space Representations of System Components
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% Position

%%%% 4th Order Approximation for Position Transfer function
% loading transfer function should load <4-D idss> ==> n4s4
% load('transfer\_function\_position\SSTF\_n4s4\_4thORDER\_ident\_fit.mat');
% SSp.OutputName = 'Zm';
% SSp.InputName = 'Zc';
% SSp.Name = 'pos. S.S. rep. (4th order, from Sys Ident Tool)';

%%% First Order Approximation for Position Transfer Function
%%% Original Position transfer function analysis.
%%% Zm/Zc = Hp = \( e^{-sTdp} \cdot \frac{wp}{s+wp} \)
%%% \( s \cdot Zm = -Wp \cdot Zm + Wp \cdot Zc \); add \( e^{-sTdp} \) to SS representation

Wp = 5.35; % Tuned frequency (rad/sec), starting guess was 0.6;

%%% Recorded Tuning Data, amplitude = 0.05, period = 5 %%%%%%%%
%%% Upward slope == 5.25 estimate lagging actual
%%% downard slope == 5.25 estimate lagging actual
%%% Upward slope == 5.5 a good match
%%% Downward slope == 5.5 estimate leading actual

SS.P.A = [-Wp];
SS.P.B = [Wp];
SS.P.C = [1];
SS.P.D = [0];
SS.P.Tdp = 0;

%%% S.S. Representation of Position
SSp = ss(SS.P.A,SS.P.B,SS.P.C,SS.P.D);
SSp.InputDelay = SS.P.Tdp;
SSp.StateName = 'Zm';
SSp.OutputName = 'Zm';
SSp.InputName = 'Zc';
SSp.Name = 'Position S.S. Representation (1st order)';

%%% Velocity State Space Representation
% Initial (old) system
%%% Vm = s*Zm*Hv = s*Zm / (s/Wv + 1); & \( sZm = -Wp \cdot Zm + Wp \cdot Zc \);
%%% \( s \cdot Vm = s \cdot zm \cdot Wv - Vm \cdot Wv \)
%%% \( = (-Wp \cdot Wv + Wp \cdot Zc + Zm) - Vm \cdot Wv \)

%%% Integrator simplification w/ "filtered derivative",
%%% Will integrate below Wv frequency, and drop info above.

%%% Vm = s*Zm
% Vam = s*Hv*Zm = s*Wv/(s+Wv)*Zm
% Vam = Wv/(1 + Wv/s) * Zm
% Vam*(1 + Wv/s) = Wv*Zm
% Vam + Vam*Wv/s = Wv*Zm
% Vam = Wv*Zm - Vam*Wv/s

% \( d(Vam/s) = -Wv \cdot Vam/s + Wv \cdot Zm \) ==> state (Vam/s)
Wv = 2*pi*5; % starting frequency 2*pi*100;

%%% Recorded Tuning Data, amplitude = 0.05, period = 5 %%%%

%% 2*pi*0.1
% Upward slope ==> severe lag
% Downward slope ==> severe lag

%% 2*pi*10
% Upward slope ==> small lead
% Downward slope ==> Small lag

%% 2*pi*4 ((less downward lag than 3, good match upward))
% Upward slope ==> good match/tiny lead
% Downward slope ==> small lag

%% 2*pi*2
% Upward slope ==> tiny lead
% Downward slope ==> small lag

%% 2*pi*3 ((3 closer than 2))
% Upward slope ==> good match
% Downward slop == small lag

% Simplified System (?Where is Vm compared to Vam?) ==> in Kalman filt.
SS.V.A = [-Wv];
SS.V.B = [Wv];
SS.V.C = [-Wv];
SS.V.D = [Wv];

%%% Simplification System
SSv.StateName = 'intVam'; %??integral of Vam
SSv.OutputName = 'Vam';
SSv.InputName = {'Zm'};
SSv.Name = 'Velocity S.S. Representation';

%%% Acceleration State Space Representation
%%%Initial (old) Equation
%%%
% Am = s*Vam = s*Vm*Wa - Am*Wa = -Wa*Wp*Wv*Zm + ... 
%
% Simplified Approximation
% Aam = s*Ha*Vm = s*Wa/(Wa+s)*Vm
% Aam = Wa/(1 + Wa/s) * Vm
%Aam*(1+Wa/s) = Wa*Vm
%Aam = Wa*Vm - Aam*Wa/s

% d(Aam/s) = -Wa*Aam/s + Wa*Vm => state (Aam/s)

V is input, A is output, same math as for velocity (from position)
Wa = 0.5*Wv;

%%% Simplified System copied from above (?Where is Vm compared to Vam?)
SS.A.A = [-Wa];
SS.A.B = [Wa];
SS.A.C = [-Wa];
SS.A.D = [Wa];
SSa = ss(SS.A.A,SS.A.B,SS.A.C,SS.A.D); %S.S. Representation of Position

%%% Simplification System
SSa.StateName = ['intAam']; %???Whats intAam??
SSa.OutputName = 'Aam';
SSa.InputName = {'Vam'};
SSa.Name = 'Acceleration S.S. Representation';
% get(SSa);

%%% Force State Space Representation
% Fltbm = Fltb * Hf
% s*Fltb = W (stochastic variable) , --> delt w/ at Kalman?
% s^2*Fltbm = -s*Fltbm*(2*dampF*Wltb) -
% Fltbm*(Wltb^2) + Fltb*(Wltb^2)

Wltb = 3927;
dampF = 0.32;
SS.F.A = [-2*dampF*Wltb -Wltb^2 Wltb^2 ; 1 0 0 ; 0 0 0]; % Wflc_noise
SS.F.B = [0 ; 0 ; 1]; %noise term added in
SS.F.C = [0 1 0]; %
SS.F.D = [0];
SS.F.Tdf = 100e-6;

%%% S.S. Representation of Force
SSf = ss(SS.F.A,SS.F.B,SS.F.C,SS.F.D);
SSf.InputDelay = SS.F.Tdf;
SSf.StateName = {'dFltbm' 'Fltbm' 'Fltb'}; %
SSf.OutputName = 'Fltbm';
SSf.InputName = 'wFltb';
SSf.Name = 'Force S.S. Representation';

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%,
%% Connecting State Space Systems Together
%% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Connecting States Together
%%% SYS = CONNECT(SYS1, SYS2, ..., INPUTS, OUTPUTS)
%% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%Final Simplification System Connections
SS.Inputs = {'Zc', 'wFltb'};
SS.Outputs = {'Zm', 'Vam', 'Aam', 'Fltbm'};
SSfull = connect(SSp, SSv, SSa, SSf, SS.Inputs, SS.Outputs);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Constructing Padé Approximation & Kalman Estimator
%% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% [KEST, L, P] = KALMAN(SYS, QN, RN, NN)
Qn = 10; % Variance of Process Noise % need to tune 1000 to 1000000 maybe?

%%%%%% Tuning of Qn %%%%%% ((while looking at force signal))
%5000 ==> too big ==> -1.#IND
%100 ==> too big ==> excessively jittery
%10 ==> smoother than original signal
%1 == > smoother than original signal, and 100
%1 == > smoother than 100, small phase delay

% [NUM, DEN] = PADE(T, N) returns the Nth-order Padé approximation
% of the continuous-time delay exp(-T*s) in transfer function form.
% The row vectors NUM and DEN contain the polynomial coefficients
% in descending powers of s.
SSfull.pade = pade(SSfull); % accounts for time delay

KalmanFull = kalman(SSfull.pade, Qn, Rn); % save 'C:\Documents and Settings\esystems\Desktop\LTB_External_Computer_Control...\
\LTB_KALMAN\KalmanFilter' KalmanFull;

disp ('Kalman Filter Construction Complete');
.7 Kalman Filter Computer

.7.1 Simulink Model

Figure 11: Kalman Filter Computer Simulink Model - Top Level
Figure 12: Kalman Filter Computer Simulink Model - Signal Scaling Block for Pos. Vel. and Acc.
Figure 13: Kalman Filter Computer Simulink Model - Signal Scaling Block Load Cell Force
Figure 14: Kalman Filter Computer Simulink Model - Kalman Filter Block
.7.2 ControlDesk Interface

Figure 15: Kalman Filter Computer ControlDesk Interface

.7.3 MATLAB Code for Initialization of Simulink Blocks

```matlab
% Creation: Chris Haller
% Kalman estimation of feedback signals

%% MATLAB Initialization
clear;
clc;
close all;

%% Load Kalman Estimator
load('LTB_KALMAN\KalmanFilter.mat');
KF.A = KalmanFull.a;
KF.B = KalmanFull.b;
KF.C = KalmanFull.c; % chic ken
KF.D = KalmanFull.d;
```
KF.StateNames = KalmanFull.StateName;

%%% Left Load Cell

% Left load cell (5000 lb)
LeftCell_geometry_scaling = 2/3;
LeftCell_gain = 5000*4.44822*1.00130;
LeftCell_zero_offset = 5+9.9;
LeftArm_mass_offset = 7.125;

% Left load cell (1000 lb)
%LeftCell_geometry_scaling = 2/3*1.00130;
%LeftCell_gain = 5000*4.44822;
%LeftCell_zero_offset = 5;
%LeftArm_mass_offset = 7.125;

% Left load cell (250 lb)
%LeftCell_geometry_scaling = 2/3*1.00130;
%LeftCell_gain = 250*4.44822;
%LeftCell_zero_offset = 6.5;
%LeftArm_mass_offset = 6.65;

%%% Right Load Cell

% Right load cell (5000 lb)
RightCell_geometry_scaling = 2/3;
RightCell_gain = 5000*4.44822*0.996359;
RightCell_zero_offset = 13.5+6.5;
RightArm_mass_offset = 7.125; % (kg) Use to calibrate zero

% Right load cell (1000 lb)
%RightCell_geometry_scaling = 2/3*0.996359;
%RightCell_gain = 5000*4.44822;
%RightCell_zero_offset = 13.5;
%RightArm_mass_offset = 7.125;

% Right load cell (250 lb)
%RightCell_geometry_scaling = 2/3*1.00450;
%RightCell_gain = 250*4.44822;
%RightCell_zero_offset = 6.5;
%RightArm_mass_offset = 7.125;

%%% Yoke, Device-Under-Test, and Gravity Constants

Yoke_mass_offset = 35.25; % (1000 and 5000 lb)
% Yoke_mass_offset = 0; %(250 lb)
DUT_mass = 1272.5; % (kg)
% DUT_mass = 0; %(250 lb)
Gravity_accel = 9.80665;

%% Acceleration, Speed, Position, Current, Voltage

% Accel
Acceleration_gain = 5.00;
Acceleration_offset = 0;

% Speed
Speed_gain = 3.0003; %(1000 and 5000 lb)
Speed_gain = 3.000; %(250 lb)
Speed_offset = +.001; %(5000 lb)
Speed_offset = 0; %(250 and 1000 lb)

% Position
Position_gain = 1.0290; %(1.00290);
Position_offset = 0; %(1000 and 5000 lb)
Position_offset = -.0007; %(250 lb)

%% Initialization Completed
disp('LTB External Control Initialization Complete')
.8 Sine and Ocean Drive Control Computer

.8.1 Simulink Model

Figure 16: Sine and Ocean Drive Computer Simulink Model - Top Level
Figure 17: Sine and Ocean Drive Computer Simulink Model - Signal Limits Check Block

Figure 18: Sine and Ocean Drive Computer Simulink Model - Limit Check Block
Figure 19: Sine and Ocean Drive Computer Simulink Model - Latch High Block

Figure 20: Sine and Ocean Drive Computer Simulink Model - Sine Wave Generator Block
8.2 ControlDesk Interface

Figure 21: Sine and Ocean Drive Computer Simulink Model - Ocean Wave Generator Block

Figure 22: Sine and Ocean Drive Computer ControlDesk Interface - Ocean Wave
.8.3 MATLAB Code

```matlab
% Creation: Chris Haller
% Initialization File for Open-Loop control of LTB
% using 'force control' Position input on LTB
% Kalman estimation of feedback signals

%% MATLAB Initialization
clear;
clc;
close all;

%% Left Load Cell
% Left load cell (5000 lb)
LeftCell_geometry_scaling = 2/3;
LeftCell_gain = 5000*4.44822*1.00130;
LeftCell_zero_offset = 5+9.9;
LeftArm_mass_offset = 7.125;

% Left load cell (1000 lb)
%LeftCell_geometry_scaling = 2/3*1.00130;
```
% Left load cell (250 lb)
LeftCell_gain = 5000*4.44822;
LeftCell_zero_offset = 5;
LeftArm_mass_offset = 7.125;

% Left load cell (250 lb)
LeftCell_geometry_scaling = 2/3*1.00130;
LeftCell_gain = 250*4.44822;
LeftCell_zero_offset = 6.5;
LeftArm_mass_offset = 6.65;

% Right load cell
RightCell_gain = 5000*4.44822*0.996359;
RightCell_zero_offset = 13.5+6.5;
RightArm_mass_offset = 7.125; % (kg) Use to calibrate zero

% Right load cell (1000 lb)
RightCell_geometry_scaling = 2/3*0.996359;
RightCell_gain = 5000*4.44822;
RightCell_zero_offset = 13.5;
RightArm_mass_offset = 7.125;

% Right load cell (250 lb)
RightCell_geometry_scaling = 2/3*1.00450;
RightCell_gain = 250*4.44822;
RightCell_zero_offset = 6.5;
RightArm_mass_offset = 7.125;

% Yoke, Device-Under-Test, and Gravity Constants
Yoke_mass_offset = 35.25; % (1000 and 5000 lb)
Yoke_mass_offset = 0; % (250 lb)
DUT_mass = 1272.5; % (kg)
DUT_mass = 0; % (250 lb)
Gravity_accel = 9.80665;

% Acceleration, Speed, Position, Current, Voltage
Accel
Acceleration_gain = 5.00;
Acceleration_offset = 0;

% Speed
Speed_gain = 3.0003; % (1000 and 5000 lb)
Speed_gain = 3.000; % (250 lb)
Speed_offset = +.001; %(5000 lb)
% Speed_offset = 0; %(250 and 1000 lb)

% Position
Position_gain = 1.00290;
Position_offset = 0; %(1000 and 5000 lb)
% Position_offset = -0.0007; %(250 lb)

%% Load Wave Dataset

%%% Data Set 1
% CSVwaveData = csvread('Elwood_wave_data\0p5m_7p5sec_Tp.txt');
% CSVwaveData1 = CSVwaveData';
% waveData.timeStep = CSVwaveData1(1,1);
% CSVwaveData2 = CSVwaveData1(1:10,2:901);
% waveData.dataValues = CSVwaveData2(:);

%%% Data Set 2
% CSVwaveData = csvread('Elwood_wave_data\0p88m_7p5sec_Tp.txt');
% CSVwaveData1 = CSVwaveData';
% waveData.timeStep = CSVwaveData1(1,1);
% CSVwaveData2 = CSVwaveData1(1:10,2:901);
% waveData.dataValues = CSVwaveData2(:);

%%% Data Set 3
% CSVwaveData = csvread('Elwood_wave_data\1p25m_7p5sec_Tp.txt');
% CSVwaveData1 = CSVwaveData';
% waveData.timeStep = CSVwaveData1(1,1);
% CSVwaveData2 = CSVwaveData1(1:10,2:902);
% waveData.dataValues = CSVwaveData2(:);

%%% Data Set 4
CSVwaveData = csvread('Elwood_wave_data\1p88m_8p8sec_Tp_pldt.txt');
CSVwaveData1 = CSVwaveData';
waveData.timeStep = CSVwaveData1(1,1);
CSVwaveData2 = CSVwaveData1(1:10,2:900);
waveData.dataValues = CSVwaveData2(:);

% Construct Time Series Wave Data Structure
waveData.length = length(waveData.dataValues); %Length of wave data set
waveData.timeValues = waveData.timeStep .* [1:waveData.length];

% Initialization Completed
disp('LTB External Control Initialization Complete')