Efficacy of Hammerstein Models in Capturing the Dynamics of Isometric Muscle Stimulated at Various Frequencies

by

Danielle S Chou

Submitted to the Department of Mechanical Engineering in partial fulfillment of the requirements for the degree of Master of Science in Mechanical Engineering at the MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 2006

© Massachusetts Institute of Technology 2006. All rights reserved.

Author........................................................................
Department of Mechanical Engineering
August 11, 2006

Certified by..............................................
Hugh M Herr
Associate Professor
Thesis Supervisor

Certified by..............................................
David L Trumper
Associate Professor
Thesis Reader

Accepted by..............................................
Lallit Anand
Chairman, Department Committee on Graduate Students
Efficacy of Hammerstein Models in Capturing the Dynamics of Isometric Muscle Stimulated at Various Frequencies

by

Danielle S Chou

Submitted to the Department of Mechanical Engineering on August 11, 2006, in partial fulfillment of the requirements for the degree of Master of Science in Mechanical Engineering

Abstract

Functional electrical stimulation (FES) can be employed more effectively as a rehabilitation intervention if neuroprosthetic controllers contain muscle models appropriate to the behavioral regimes of interest. The goal of this thesis is to investigate the performance of one such model, the Hammerstein cascade, in describing isometric muscle force. We examine the effectiveness of Hammerstein models in predicting the isometric recruitment curve and dynamics of a muscle stimulated at fourteen frequencies between 1 to 100Hz. Explanted frog plantaris longus muscle is tested at the nominal isometric length only; hence, the muscle’s force-length and force-velocity dependences are neither assumed nor ascertained. The pilot data are fitted using ten different models consisting of various combinations of linear dynamics with polynomial nonlinearities. Models identified using data with input stimulation frequencies of 20Hz and lower generate an average RMS error of 12% and are reliably stable (87 of 90 simulations). Between 25 to 40Hz, the average error generated by the estimated models is 10%, but the estimated dynamics are less stable (16 of 30 simulations). Above 40Hz, linear and Hammerstein nonlinear models fail to consistently generate stable dynamic estimates (11 of 30 simulations), and errors are large (\(\bar{e}_{RMS} = 44\%\)). Simulations also suggest Hammerstein models found iteratively do not perform much better than linear dynamic systems (in general, \(\bar{e}_{RMS} = 10 \text{ to } 15\%\)). In addition, simulations using iterated nonlinearities generate RMS errors that are comparable to those simulations using a fixed nonlinearity (both about 16%). These preliminary results warrant further investigation into the limits of Hammerstein models found iteratively in the identification of isometric muscle dynamics.

Thesis Supervisor: Hugh M Herr
Title: Associate Professor

Thesis Reader: David L Trumper
Title: Associate Professor
0.1 Acknowledgments

To the members of Biomechatronics, particularly Max and Sam, who helped provide structure and focus when I began to feel lost, and Waleed, who answered my many questions and toughened me up; Biomech’s veterinarians, Katie and Chris, for their hours of dissection; Tony, Angela, Sungyon, Tina, and the rest of the Perfect Strangers, the people on whom I’ve unburdened myself most in the past two years; and last but certainly not least, Mom, Dad, and Jeff for always being there when I felt I had no one—thank you all for helping me stay sane in your various capacities.
Contents

0.1 Acknowledgments ........................................... 5

1 Introduction ............................................. 17
  1.1 Objectives ........................................... 20
  1.2 Thesis organization ................................... 20

2 Background .............................................. 23
  2.1 Motor unit physiology .................................. 23
     2.1.1 Neuron physiology ................................ 23
     2.1.2 From neurons to muscle fibers ...................... 25
     2.1.3 Muscle fibers and the actin-myosin complex ...... 26
     2.1.4 Muscle feedback sensing ................................ 28
  2.2 Muscle behavior .................................... 28
     2.2.1 Summation of forces ................................ 29
     2.2.2 Muscle recruitment .................................. 30
     2.2.3 Non-isometric muscle contraction .................... 32
  2.3 Muscle models ........................................ 34
     2.3.1 Hill-type models .................................. 36
     2.3.2 Biophysical models .................................. 39
     2.3.3 Mathematical models ................................ 40
     2.3.4 Hammerstein model .................................. 41

3 Methods and Procedures .................................. 47
  3.1 Hardware ............................................ 47
3.1.1 Experimental set-up ........................................ 47
3.1.2 Muscle set-up ............................................. 49
3.2 Experimental procedure ..................................... 50
3.3 System identification and simulation ..................... 51
  3.3.1 The static input nonlinearity ......................... 52
  3.3.2 The linear dynamic system ........................... 55
  3.3.3 Bootstrapping .......................................... 57
  3.3.4 Simulation order ...................................... 59

4 Results and Discussion ....................................... 61
  4.1 A prototypical system identification .................... 62
  4.2 Is the input pulse-width range adequate to reach force saturation?  65
  4.3 Can the algorithm identify the appropriate recruitment curve?  67
  4.4 What basis functions should be used to fit the input nonlinearity? 67
  4.5 What is the appropriate input delay? ........................ 68
  4.6 How does a dynamic system with no input nonlinearity perform? 72
  4.7 How do different combinations of input polynomial nonlinearities and
dynamic systems perform? .................................. 78
  4.8 How does a system with fixed input nonlinearity perform? .......... 81
  4.9 How does a system with fixed input nonlinearity and fixed dynamics
perform for high stimulation frequency input? .................. 83
  4.10 Comparison of all simulation results ...................... 86
    4.10.1 Simulation errors .................................... 86
    4.10.2 Simulation stability ................................. 90

5 Conclusion ..................................................... 95
  5.1 Future work ................................................. 96

A Experimental Protocol ....................................... 97

B MATLAB code .................................................. 99
**List of Figures**

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2-1</td>
<td>Neuron physiology</td>
<td>25</td>
</tr>
<tr>
<td>2-2</td>
<td>Sarcomere</td>
<td>26</td>
</tr>
<tr>
<td>2-3</td>
<td>Actin-myosin cycle from <a href="http://www.essentialbiology.com">http://www.essentialbiology.com</a></td>
<td>27</td>
</tr>
<tr>
<td>2-4</td>
<td>Isometric twitch and tetanus</td>
<td>29</td>
</tr>
<tr>
<td>2-5</td>
<td>Force-length relationships for pennate-fibered gastrocnemius and parallel-fibered sartorius</td>
<td>33</td>
</tr>
<tr>
<td>2-6</td>
<td>Force-velocity relationship and the underlying actin-myosin mechanism</td>
<td>35</td>
</tr>
<tr>
<td>2-7</td>
<td>Surface created by force-length and force-velocity relationships</td>
<td>35</td>
</tr>
<tr>
<td>2-8</td>
<td>Hill-type model used by Durfee and Palmer (1994)</td>
<td>37</td>
</tr>
<tr>
<td>2-9</td>
<td>Hill-type model used by Ettema and Meijer (2000)</td>
<td>38</td>
</tr>
<tr>
<td>2-10</td>
<td>Hill-type rheological model used by Forcinito, Epstein, and Herzog (1998)</td>
<td>38</td>
</tr>
<tr>
<td>2-11</td>
<td>Hammerstein model structure</td>
<td>41</td>
</tr>
<tr>
<td>3-1</td>
<td>Experimental setup</td>
<td>48</td>
</tr>
<tr>
<td>4-1</td>
<td>Hammerstein model structure</td>
<td>62</td>
</tr>
<tr>
<td>4-2</td>
<td>Example identification after 1 and 2 iterations (2Hz data)</td>
<td>63</td>
</tr>
<tr>
<td>4-3</td>
<td>Example identification after three, four, and twelve iterations (2Hz data)</td>
<td>64</td>
</tr>
<tr>
<td>4-4</td>
<td>Isometric recruitment relationship - 1Hz</td>
<td>66</td>
</tr>
<tr>
<td>4-5</td>
<td>Isometric recruitment relationship - 2Hz</td>
<td>66</td>
</tr>
<tr>
<td>4-6</td>
<td>Delay selection: 3Hz data</td>
<td>69</td>
</tr>
<tr>
<td>4-7</td>
<td>Delay selection: 7Hz data (continued on next page)</td>
<td>70</td>
</tr>
<tr>
<td>4-8</td>
<td>Delay selection: 7Hz data (continued from previous page)</td>
<td>71</td>
</tr>
<tr>
<td>4-9</td>
<td>Simulations with no nonlinearity: muscle response at 3Hz stimulation</td>
<td>73</td>
</tr>
</tbody>
</table>
4-10 Simulations with no nonlinearity: muscle response at 40Hz stimulation 74
4-11 Simulations with no nonlinearity: muscle response at 100Hz stimulation (all have unstable dynamics) 75
4-12 Simulation errors with no nonlinearity 76
4-13 Difference equation denominator coefficient variation 77
4-14 Simulation errors of different dynamic systems with different nonlinearities (in scatter and bar graph format) 79
4-15 Comparison of second- and third-order dynamic fits (2Hz data identified iteratively with third-order nonlinearity and five delays) 80
4-16 Simulation errors for fixed and iteratively-found recruitment nonlinearity (third-order dynamics, third-order polynomial) 82
4-17 Difference equation denominator coefficient variation 82
4-18 Muscle response estimates for high stimulation frequency input using fixed nonlinearity and fixed dynamics trained from 2Hz data 84
4-19 Simulations with fixed third-order polynomial nonlinearity and third-order dynamics 85
4-20 Summary of simulation errors (scatter and bar graphs) 87
4-21 Average simulation errors across stimulation frequencies 88
4-22 Average model errors across seven data sets below 25Hz 88
4-23 Pole-zero maps for third-order dynamic systems with different input polynomial nonlinearities 91
4-24 Fluctuating passive muscle force: 1Hz data 92
4-25 Average stability of ten estimated models across stimulation frequency (1 = stable, 0 = unstable) 92

C-1 No nonlinearity, second-order dynamics (Experiments 1 to 8: 1, 20, 3, 50, 10, 75, 15, 100Hz) 108
C-2 No nonlinearity, second-order dynamics (Experiments 9 to 15: 7, 30, 1, 5, 40, 25, 2Hz) 109
<table>
<thead>
<tr>
<th></th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>C-3</td>
<td>No nonlinearity, third-order dynamics (Experiments 1 to 8: 1, 20, 3, 50, 10, 75, 15, 100Hz)</td>
<td>110</td>
</tr>
<tr>
<td>C-4</td>
<td>No nonlinearity, third-order dynamics (Experiments 9 to 15: 7, 30, 1, 5, 40, 25, 2Hz)</td>
<td>111</td>
</tr>
<tr>
<td>C-5</td>
<td>No nonlinearity, fourth-order dynamics (Experiments 1 to 8: 1, 20, 3, 50, 10, 75, 15, 100Hz)</td>
<td>112</td>
</tr>
<tr>
<td>C-6</td>
<td>No nonlinearity, fourth-order dynamics (Experiments 9 to 15: 7, 30, 1, 5, 40, 25, 2Hz)</td>
<td>113</td>
</tr>
<tr>
<td>C-7</td>
<td>Difference equation denominator coefficient parameter variation</td>
<td>114</td>
</tr>
<tr>
<td>C-8</td>
<td>Nonlinear recruitment curves from iterative identification with third-order dynamics (Experiments 1 to 9: 1, 20, 3, 10, 15, 7Hz)</td>
<td>115</td>
</tr>
<tr>
<td>C-9</td>
<td>Nonlinear recruitment curves from iterative identification with third-order dynamics (Experiments 11 to 15: 1, 5, 40, 25, 2Hz)</td>
<td>116</td>
</tr>
<tr>
<td>C-10</td>
<td>Simulations with fixed third-order polynomial nonlinearity and third-order dynamics (Experiments 1 to 6: 1, 20, 3, 50, 10, 75Hz)</td>
<td>117</td>
</tr>
<tr>
<td>C-11</td>
<td>Simulations with fixed third-order polynomial nonlinearity and third-order dynamics (Experiments 7 to 12: 15, 100, 7, 30, 1, 5Hz)</td>
<td>118</td>
</tr>
<tr>
<td>C-12</td>
<td>Simulations with fixed third-order polynomial nonlinearity and third-order dynamics (Experiments 13 to 15: 40, 25, 2Hz)</td>
<td>119</td>
</tr>
<tr>
<td>C-13</td>
<td>Pole-zero maps for second-order dynamic systems with different input polynomial nonlinearities</td>
<td>120</td>
</tr>
<tr>
<td>C-14</td>
<td>Pole-zero maps for third-order dynamic systems with different input polynomial nonlinearities</td>
<td>121</td>
</tr>
<tr>
<td>C-15</td>
<td>Pole-zero maps for fourth-order dynamic systems with different input polynomial nonlinearities</td>
<td>122</td>
</tr>
</tbody>
</table>
List of Tables

4.1 Average linear model RMS errors across seven data sets below 25Hz . 89

C.1 RMS errors for all simulations . . . . . . . . . . . . . . . . . . . . . . 123

C.2 Stability for all simulations . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 124
THIS PAGE INTENTIONALLY LEFT BLANK
Chapter 1

Introduction

Today more than five million Americans suffer from stroke [1] with the associated side-effects of movement impairment. An additional 250,000 Americans are living with spinal cord injuries [40]. With such a population in potential need of movement aids, research has intensified in the field of neuroprosthetics and functional electrical stimulation (FES) control.

FES is used in cases when the skeletal muscle tissues are intact, but their neural activation has been impaired due to nervous tissue damage resulting from traumatic injury or disease. When such pathologies occur, an external electrical stimulation can be applied to a motoneuron to generate movement. The electrical signal depolarizes the nerve, and an action potential propagates down the nerve to the neuromuscular junction. The muscle then becomes depolarized, resulting in calcium release from the sarcoplasmic reticulum. The calcium release triggers the interaction of actin and myosin proteins and causes the muscle to generate force.

FES controllers contain muscle models that allow them to determine what input would be appropriate to elicit a particular muscle behavior. For demonstrations of FES control, it is not imperative that a prosthesis controller use an accurate muscle model. As shown by multiple researchers [46, 9, 14, 10], feedback controllers require little knowledge of the muscle system in advance, yet are still able to modulate force. However, this brute force method requires constant re-stimulation of muscle to achieve the desired force and would be unsuitable for any realistic long-term prosthesis. In-
stead, a good muscle model should help minimize the amount of stimulation to which we subject the muscle, thereby preventing fatigue and tissue damage. In better understanding the relationship between electrical activation and muscle force behavior, it is likewise possible to better and more effectively employ neural prostheses for the restoration of movement in a paralyzed limb.

The better the model linking input stimulation to force output, the better the control exerted by a neural prosthesis. This search for a sufficiently descriptive model has driven past and ongoing research into muscle modeling. One of the more popular muscle models has been the nonlinear Hammerstein system. The Hammerstein model consists of a static input nonlinearity followed by linear dynamics. It is popular because it readily lends itself to a physiological analogy: the input nonlinearity may be interpreted as muscle recruitment and the linear dynamics as the dynamics of cross-bridge formation.

Beginning in 1967, Hammerstein models have been used in practical prostheses. That year, Vodovnik, Crochetiere, and Reswick [46] developed a closed-loop controller for an elbow prosthesis. Feeding back position, their controller linked elbow angle to input voltage via joint torque and contractile force. The model accounted for recruitment voltage threshold and saturation by using a piecewise linear recruitment nonlinearity.

In 1986, Bernotas, Crago, and Chizeck [4, 27] used a Hammerstein system to model isometric muscle. They explored force dependence on muscle length and stimulation frequency and found the relationships were different for different muscles depending on slow- versus fast-twitch fiber composition. Though at the time they did not use their estimated model as part of a controller, the use of recursive least squares estimation meant online identification was possible.

Two years later, Chizeck, Crago, and Kofman [9] created a closed-loop muscle force controller. The pulse-width modulated controller included a muscle model comprised of a memoryless nonlinearity in series with a continuous second-order dynamic system. The group demonstrated that though their controller did not require a precise model of muscle dynamics, it was still able to regulate force while maintaining stability and
repeatability.

In 1998, Bobet and Stein [6] modeled dynamic isometric contractions with a modified Hammerstein model. Dynamic in their case meant an input modulated by pulse rate rather than pulse-width, and the Hammerstein modification consisted of a linear filter preceding the traditional Hammerstein cascade. In addition, the output force dynamics depended on a force-varying rate constant $b$. Inputting the same pulse train, estimated force profiles were compared to experimental force measurements; results showed that unfused and fused tetanus were well-simulated though isometric twitch was not. No active controller was attempted though the authors suggested that for an adaptive controller of isometric muscle, only one parameter—the force-varying rate constant $b$—need be adjusted.

In 2000, Munih, Hunt, and Donaldson [38] used a Hammerstein model-based controller to help normal and impaired subjects maintain balance. After obtaining recruitment curves based on Durfee and MacLean’s twitch-response method [15], Munih et al. then had subjects stand rigidly on a wobbling platform. The patients’ bodies thus acted essentially as inverted pendulums, and balancing torques were achieved by stimulation of ankle plantarflexors.

Because of its ready translation into a well-established physiological phenomenon and its relative speed in identification compared to other mathematical models, the Hammerstein-based isometric model has been employed time and time again throughout the decades of research into FES-motivated muscle models. However, the model has its shortcomings. The foremost two flaws are its assumption of linear dynamics and its disregard of any recruitment-dynamics coupling. Recruitment and dynamics are coupled in a manner described by Henneman’s principle: small neurons innervate slow-twitch fibers, and large neurons innervate fast-twitch fibers. Thus, natural muscle recruitment, which begins with the smallest and progresses to the largest neurons, affects dynamics by recruiting the slowest to fastest muscles [25]. Yet, the Hammerstein model continues to be used. The question then arises, to what extent is the model appropriate in describing the force behavior of pulse-width modulated isometric muscle? And what are the limits for the identification process?
1.1 Objectives

When is it appropriate to use a Hammerstein model to describe pulse-width modulated isometric muscle force? What are the limits to the identification of such a model? This thesis seeks to address these questions in one aspect: that of stimulation frequency. It aims to assess the efficacy of a Hammerstein model in iterative identification of muscle undergoing a range of stimulation frequencies between 1 and 100Hz. Specifically, it attempts to determine the maximum stimulation frequency at which a Hammerstein model identified iteratively may still accurately describe muscle behavior. Only isometric muscle contraction is considered; while muscle clearly does not operate only in this regime, this simplification allows the use of a traditional Hammerstein model with no consultation of muscle length or shortening velocity. Different combinations of dynamics and polynomial nonlinearity are tested. The best combination is then simulated across all data sets with each data set allowed to choose and validate its own dynamic system and regressor coefficients. Simulations are also run for each data set with a fixed input nonlinearity and dynamics chosen by MATLAB’s oe command. Errors and dynamic system parameter variation are compared across experiments run with different stimulation frequencies. Stability of the estimated systems and model iterative convergence are also examined.

1.2 Thesis organization

Chapter 2 reviews muscle physiology, then describes the different types of muscle models popular in the literature. The structure of the model of interest, the Hammerstein cascade, is discussed in greater detail along with methods of identifying Hammerstein systems and convergence of the iterative method. Chapter 3 begins with a description of the experimental hardware, followed by a discussion of the procedures and simulation methods. Chapter 4 presents the simulation results and attempts to explain them in the context of muscle mechanics. Chapter 5 summarizes the results and ends with potential future work and future questions. Appendix A contains experiment
notes, Appendix B contains the MATLAB iterative code used to run simulations, and Appendix C is comprised of simulation data.
Chapter 2

Background

This chapter contains background on the relevant biology and modeling history essential to exploring the use of Hammerstein models in muscle identification. A brief description of muscle physiology and motor units is given, followed by §2.2 on muscle phenomenological behavior. The chapter then embarks on a review of different approaches to muscle modeling and concludes with a summary of the history and issues encountered specifically by Hammerstein muscle models.

2.1 Motor unit physiology

Most muscle models are steeped in understanding of the underlying physiological mechanics behind muscle innervation. Hence, this section discusses two components comprising a single motor unit: a motoneuron and the multiple muscle fibers it can innervate. Both are fundamental to understanding muscle contraction.

2.1.1 Neuron physiology

A motoneuron consists of a soma (cell body), a long axon, and dendrites branching from the cell body, as shown in Figure 2-1. Each motor neuron has only one axon, but the axon may branch into a maximum of 100 axon terminals. Synapses connect the transmitting axon terminals of one neuron to either the receiving dendrites of
adjacent neurons or the motor end plates of multiple muscle fibers. A motor end plate is an area of muscle cell membrane that contains receptors for the excitatory neurotransmitter, acetylcholine.

Weak signals from the individual dendrites collect near the base of the axon at a location called the axon hillock until a temporal and/or spatial integration of the signals results in a voltage exceeding the threshold voltage (-55mV). At that point, the neuron depolarizes and an action potential propagates down the axon to the axon terminal where it will release neurotransmitter molecules. The neurotransmitters cross the synapse and temporarily bind to the corresponding neurotransmitter receptors of the next neuron. This signal continues from one neuron to the next until it reaches a neuron whose axon is connected to a motor end plate. An axon may contact several muscle fibers, but each muscle fiber is innervated by a single axon. That single axon and the muscle fibers it contacts form a motor unit (Kandel et al. [32, pp 28-31]).

An action potential occurs when the sodium ion channels open. The behavior of ion channels also explains why action potentials may only travel in one direction, from dendrite to axon terminal: the ion channels have refractory periods of several milliseconds during which they cannot open or close, no ions can travel in or out of the axon, and therefore the voltage at that point along the axon cannot change. Refractory periods partly dictate conduction velocities. Signal conduction velocities also vary with the diameter of the neuron; small neurons conduct more slowly than large ones. In addition, conduction speed is effected by nerve myelination: nerves coated with the fatty insulating substance myelin have much faster conduction velocities (4 to 120 m/s) than unmyelinated nerves (0.4 to 2.0 m/s) (Kandel et al. [32, p 678]). When studying muscle mechanics, nerve conduction speeds and delays are relevant as they have an effect on muscle system dynamics.

Neuron input can be either excitatory (depolarizing in which axon voltage becomes less negative) or inhibitory (hyperpolarizing in which axon voltage becomes more negative) depending on the type of neurotransmitter. For muscle, acetylcholine (ACh) is the neurotransmitter. Signals received through dendrites closer to the axon hillock are weighted more heavily in the decision to send an action potential.
2.1.2 From neurons to muscle fibers

At the terminus of a motoneuron, the neurotransmitter acetylcholine is released. Muscle fibers on the other side of the synapse have acetylcholine receptors, and when the receptors bind acetylcholine molecules, they cause sodium ion channels to open. An action potential spreads via the transverse tubules (T-tubules) within the muscle fiber, and the muscle fiber sarcoplasm (cytoplasm) depolarizes. As a result of the depolarization, calcium ion channels in the muscle fiber’s sarcoplasmic reticulum open, and calcium ions diffuse into the sarcoplasm. These ions are key to the interaction of
muscle fibers' actin and myosin filaments, which brings us to the next section.

2.1.3 Muscle fibers and the actin-myosin complex

Muscle fibers contain clusters of sarcomeres, the basic contractile unit. A sarcomere is 1.5-3.5 μm long and consists of three primary components as shown in Figure 2-2: the thin actin filament (in green), the thick myosin filament (in red), and the Z-disks defining the length of a sarcomere when they are arranged end on end to form a muscle fiber. Myosin heads stick out from the myosin filament along the length of the myosin filament except for a region in the middle. Several theories have been put forth regarding muscle contraction, but the widely accepted theory was proposed by A.F. Huxley in 1957. He speculated that unbonded myosin heads are in a cocked position, as shown in Figure 2-3(a), due to a molecule of ADP attached to the myosin head.
Meanwhile, an action potential causes calcium ions to be released by the sarcoplasmic reticulum, and these ions bond to parts of the actin filament (see Figure 2-3(b)). In so doing, the actin filament's conformation has been altered and bonding sites are exposed. The myosin heads attach to these exposed sites to form cross-bridges, which have a different preferred structure that requires the myosin head to rotate while releasing the molecule of ADP (see Figure 2-3(c)). As the myosin head rotates, the overlap between the actin and myosin filaments increases and results in the overall length of the sarcomere shortening (≈ 0.06 μm). This is called the cross-bridge power stroke. To release the bonds between the myosin and actin filaments at the end of the power stroke, ATP is required. It binds to the myosin head, which then detaches from the actin filament and the muscle fiber relaxes (see Figure 2-3(d)). Subsequently, the ATP is dephosphorylated (releases a phosphate molecule) to become ADP, which once again cocks the myosin head for future attachment to another actin binding site (Kandel et al. [32, p 678], see also [31, 24]).

![Actin-myosin cycle](http://www.essentialbiology.com)

Figure 2-3: Actin-myosin cycle from http://www.essentialbiology.com

If ATP or calcium ions are lacking, such as when an animal dies, rigor mortis sets in because the released ADP can no longer be phosphorylated to become an
ATP molecule. Thus, the cross-bridges cannot be broken and the muscles remain contracted.

Muscle fibers come in two types: fast-twitch white (fast-fatigue) and slow-twitch red (fatigue-resistant) fibers. White and red fibers metabolize oxygen and glucose differently. Depending on the species and the muscle in question, the composition of red and white fibers will vary. Fiber composition affects muscle dynamics and thus dictates the system time constants.

Muscle contraction dynamics are not only dependent on fiber composition and voltage activation, they are also dependent on length, shortening velocity, and tension history. These dependences are suggested by physiological evidence of muscle feedback sensing, which leads to the next subsection.

### 2.1.4 Muscle feedback sensing

Learning what properties muscles use for feedback allows biomechanists to determine what states are relevant in their modeling and, ultimately, for control. Muscle feedback sensing is conducted via a number of sensory neurons that originate in skeletal muscle. Of the six types, three are directly related to muscle contraction mechanics. These sensory organs are the primary spindle endings (type Ia), which sense muscle length and strain rate; golgi tendon organs (type Ib), which sense tension; and secondary spindle endings (type II), which sense muscle length. Dysfunctions in any of these sensing organs can result in abnormal system dynamics such as spasticity and oscillatory tremor [52, 43].

Physiological feedback sensing hints at what states affect muscle behavior, but what are these behaviors? The next section discusses three dominant behaviors in some detail.

### 2.2 Muscle behavior

As a result of its underlying microscopic mechanisms, muscle exhibits certain macroscopic behaviors. For example, sarcomere length dictates muscle's force-length rela-
tionship, and fiber composition can affect recruitment. This section details three specific behaviors: force summation, muscle recruitment, and force-length and -velocity dependences. The latter, though not pertinent to this thesis' study of isometric contraction, is crucial to the understanding of muscle in different behavioral regimes and would need to be accounted for by any practical neuroprosthetic controller. We begin with the first behavior, force summation.

2.2.1 Summation of forces
For muscle, a single action potential causes an isolated twitch contraction analogous to a system impulse response, as shown in Figure 2-4(a). A twitch results in relatively little force being produced by the muscle because the action potential does not release enough calcium ions to form a considerable number of cross-bridges, and the number of cross-bridges is directly proportional to the force produced.

When action potentials occur more often, cross-bridge formation will increase, and individual twitches will overlap and begin to add (see Figures 2-4(b) and 2-4(c)). This is called unfused tetanus. Eventually, when the calcium ion release rate is greater...
than the rate at which the ions are reuptaken, a fused tetanus will occur in which a constant force is sustained over a period of time, as illustrated in Figure 2-4(d).

Fiber length also affects the maximum force produced. With an increased number of cross-bridges, more overlap between filaments will occur. If the fibers are very short, then the amount of permissible overlap quickly gets saturated and no additional force may be produced. Fiber length, too, varies across species, and a single muscle may have non-uniform muscle fiber lengths in different sections of the muscle belly. In muscle modeling, the non-uniformity is almost always neglected in an effort to simplify the modeling problem.

Overlapping twitch responses is not the only way to generate forces greater than that of a single twitch. It is also possible to generate higher forces by contracting multiple muscles. This is done via muscle recruitment.

### 2.2.2 Muscle recruitment

A single muscle fiber is able to produce only a limited amount of force. For higher force production to occur, multiple muscle fibers must be recruited. In vivo, muscle fiber recruitment begins with the smallest motoneurons and gradually increase to larger ones. According to Henneman's principle [25], this means slow-twitch fibers are recruited before fast-twitch ones. This is metabolically more efficient and is why animals can demonstrate very fine motor control. In contrast, experimental stimulation generally is unable to follow this recruitment sequence. Current electrode technology remains relatively crude and bulky when compared to the size scale of nerves and muscle fibers, so it has been difficult electrically stimulating anything but the largest motoneurons. Thus, when muscle recruitment is mentioned in the context of functional electrical stimulation (FES), it generally refers to the number of active muscle fibers rather than the increasing size of the motoneurons recruited.

Muscles can be recruited either via spatial or temporal summation. Physiologically, this hearkens back to the cross-bridge theory: either the area of stimulation is increased or stimulation is sustained in order to increase the concentration of calcium ions required to form cross-bridges. Spatial recruitment via pulse amplitude or pulse
period is more common, but some researchers have explored temporal recruitment [6, 10].

The isometric recruitment curve (IRC) may be obtained a few different ways. Durfee and MacLean [15] suggested four, the first three of which they discussed in great detail: step response, peak impulse response, ramp deconvolution, and stochastic iteration [39, 29]. Each have their advantages and disadvantages, as discussed below.

- Step response methods may fatigue the muscle but make no assumptions about linear dynamics.

- Peak impulse response experiments, on the other hand, fatigue muscles less but rely on assumed linear dynamics.

- Ramp deconvolution, the authors' proposed method, requires an estimate of system dynamics based on an averaged impulse (twitch) response; the measured force is then deconvolved with the impulse response to get the recruitment curve. However, since deconvolution amplifies noise, a noise-free dynamic estimate was enforced by restricting the impulse response to be a critically damped second order system with a double real pole. This assumption allowed very rapid identification of the IRC.

- The iterative process makes no assumptions as to the form of the nonlinearity or the dynamics, but it is not guaranteed to converge and can be computationally expensive.

Durfee and MacLean concluded that each method resulted in slightly different curves, but no one method was more correct than the others: method selection should vary with the factors most important to the experiment, such as muscle fatigue or identification time.
2.2.3 Non-isometric muscle contraction

The previous two subsections described ways of generating forces greater than that of a single muscle twitch. Yet, those are not the only two factors contributing to muscle's force output. The magnitude of force is also dependent on length and velocity; a muscle with changing length behaves differently than a muscle whose length is fixed. Though this thesis deals exclusively with isometric muscle stimulation, a brief review of non-isometric behavior is included here to highlight the potential complexity demanded by an all-encompassing muscle model. However, limiting the scope of the muscle identification problem to isometric contraction is not a wasted exercise. As others have pointed out, muscles whose series elasticity approaches infinite stiffness will have output behavior approaching that of isometric muscle [5], and the isometric model may be a foundation upon which to build non-isometric ones [19].

We begin with a description of the force-length relationship and follow with a discussion on the force-velocity relationship.

**Force-length relationship**

Muscle fiber force depends on sarcomere length and, correspondingly, muscle fiber length. Force is also affected by the configuration of the muscle; that is, whether the muscle fibers are exactly parallel to the axis of the muscle or at an angle (pennate). Pennate muscles, due to their fiber configuration, allow more muscle fibers to work in parallel, so the forces generated are usually larger [36].

Figure 2-5, from McMahon's book [36], illustrates force-length relationships for two different types of muscles, gastrocnemius and sartorius. The gastrocnemius, a large calf muscle, is composed of short, pennate fibers while the sartorius, which is a thin and very long thigh muscle, is comprised of parallel fibers. Three curves are shown for each muscle: passive force occurs when the muscle is simply stretched but not contracting; active force, labeled 'Developed,' accounts only for contractile force generation; and total force is a summation of both active and passive contributions. The force-length relationship describes the steady-state force produced by
cross-bridges when muscle length is constant, i.e., an isometric response. At low strains, the active force is low because the number of cross-bridges is low. As the strain increases, the number of cross-bridges increases and therefore isometric force increases despite overlap between the thin actin filaments. At yet higher strain rates, the peak active force is produced and plateaus slightly because the middle region of the thick myosin filament lacks myosin heads, as discussed in §2.1.3. This strain is often termed $L_0$ and corresponds to muscle rest length. At rest length, peak power is produced. Straining further results in a steady decline in force as the overlap between thick and thin filaments decreases. This decline is further accelerated when the thick filaments collide with the bounding Z-discs, and a force against the contractile strain is produced (Kandel et al. [32, p 682]).

Meanwhile, passive forces are also at work. Sarcomeres are active components, but there are also several passive elements. Tendon is one such element, and it can significantly affect muscle force behavior. In addition to tendon, connecting filaments attach the thick myosin filaments to the Z-discs at each end of a sarcomere, while connective tissue surrounds each muscle fiber to aid in even distribution of force and strain along the entire muscle fiber length. Passive components can contribute a considerable amount of force depending on the muscle strain. At low strains, the connecting filaments and tissue exert no force as they are not yet in tension. At
approximately $L_0$, the passive elements begin to produce a force that increases exponentially with strain until at very high strains, the force produced is entirely due to passive components. This can be seen in Figure 2-5.

**Force-velocity relationship**

Besides muscle’s force-length dependence, it also exhibits force-velocity dependence. For muscles, contraction velocity is proportional to the rate of energy consumption. During muscle shortening when the contractile force exceeds external load and concentric work is done, the velocity is positive and high but there is little force produced because the myosin heads must rotate to be recocked for future cross-bridge formation. In contrast, during muscle lengthening when the external load exceeds contractile forces and eccentric work is done, the velocity is negative and force output is high because the myosin heads do not need to be recocked. Instead, as the actin filament slides past, the head rotates past its neutral location. When the bond is released, the head merely rotates back to its neutral location where it is again ready for cross-bridge formation with another actin binding site. Figure 2-6 illustrates this action.

The force-velocity relationship acts independently of the force-length relationship. Together they form a surface of muscle behavior shown in Figure 2-7.

2.3 Muscle models

Any sort of muscle control requires a working model of the muscle to be controlled, and the better the model, the more efficiently functional electrical stimulation can elicit desired behavior. There are many types of models, but they may be lumped into three categories: Hill-type mass-spring-damper models, Huxley-based biophysical models, and what some call the “model-free” purely mathematical model [12].

No one muscle model is perfect; all of them make simplifications, and each has its advantages and disadvantages. While biophysical models seek to explain macroscopic-scale behavior through microscopic-scale mechanics, they are generally not compu-
Figure 2-6: Force-velocity relationship and the underlying actin-myosin mechanism

Figure 2-7: Surface created by force-length and force-velocity relationships

...tationally tractable, especially when applied to practical controllers that may be implemented on mobile prosthetic and orthotic systems. Meanwhile, macroscopic-scale models often generalize based on isometric observations or Hill-type relationships and end up missing behaviors that are unique to muscle’s nonlinearities. Amongst these features of muscle which scientists and engineers try to explain are:
the tendency to fatigue over time depending on a particular muscle's white (fast-fatigue) versus red (fatigue-resistant) fiber composition;

- the ability to potentiate, i.e., produce more than double the force output of one single pulse when the muscle is subjected to two rapid stimulations in succession;

- the catch-like effect, similar to potentiation but longer-lasting, in which an extra pulse during a regular pulse train results in lasting force enhancement. This is especially evident in fatigue-resistant muscle; and

- stretch-induced force enhancement and shortening-induced force depression.

A potential solution to modeling these behaviors is using a purely mathematical model. However, purely mathematical models can be free of any physical mechanism, it may be highly difficult to gain intuition that would enable better understanding and prediction of muscle behavior.

Though this thesis addresses one specific example from the last category of models, a few examples from each category are reviewed so as to illustrate the breadth of muscle modeling and the alternatives to the chosen Hammerstein model. For a more comprehensive history and review of muscle models, please consult [24, 44]. A literature review of Hammerstein muscle models concludes this chapter.

### 2.3.1 Hill-type models

Hill-type models are by far the most popular of the three categories due to their relative simplicity and their ability to be analyzed by classical mechanical methods. These models consist of masses, springs, dampers, and black-box contractile elements and generally concern themselves with describing force-length and -velocity dependences. A.V. Hill's original model [26] was comprised of a spring in series with a contractile element; the contractile element is a force generator whose behavior is described by what has become the rudimentary hallmark of force-velocity relationships:

\[(F + a) \cdot v = (F_0 - F) \cdot b\]
Most Hill-type models are variations of the contractile element with springs and dampers in series and/or parallel. For example, Durfee and Palmer used a contractile element in series with a nonlinear spring (active elements) and both in parallel with a second nonlinear spring and a nonlinear dashpot (passive elements). The total force output of the muscle is modeled as a sum of the forces from active and passive elements. [16], as shown in Figure 2-8. Physiologically, the model is a mechanical decomposition meant to imitate the cross-bridge force production mechanism in series with tendon. Length and velocity-dependences are ascribed to the nonlinear spring and damper. The mass-spring-damper system is what frequently gives rise to muscle’s description as a second order dynamic system.

Ettema and Meijer chose a different configuration of Hill-type models. They compared three models, one of them a Hill-type model consisting of a contractile element with a series spring and both in parallel with another spring (see Figure 2-9); the second, a modified Hill model whose contractile element has a contraction history dependence; and a third called the exponential decay model. This last is a third order linear model with no explicit force-velocity relationship; instead, its contractile element acts based on both contraction history and force-length history. Previous changes in length exponentially decay with time [17]. Like many researchers, Ettema and Meijer tailored their models to capture a specific behavior: in this case, stretch-induced force enhancement and shortening-induced force depression for subsequently
Forcinito, Epstein, and Herzog [20] also sought to duplicate non-isometric force enhancement and depression using a Hill-based model. They chose what they called a rheological model, as shown in Figure 2-10(c); it is distinct mechanically in its employment of an elastic rack-and-pinion type of setup that allows the stiffness of the model to vary as a function of muscle length. This parallel elastic element is engaged only when the muscle is activated. Again, Hill-type contractile elements, springs, and dampers are present. In this case, Hill's force-velocity relationship is not embedded.
in the contractile element but rather falls out of the combined behavior of the me-
chanical components. At the end of their discussion, Forcinito et al emphasize that
they make no attempt to provide an all-compassing muscle model covering different
behavioral regimes; rather, they point out that many simple models can represent
different aspects of muscle behavior.

2.3.2 Biophysical models

Hill-type models are useful for gaining intuition about muscle as mass, spring, and
damper-like systems, but the models are purely phenomenological; that is, they are
based only on output behavior and make no reference to the relatively well-understood
underlying cross-bridge mechanics causing the behavior. Some researchers have de-
developed models based on Andrew Huxley's 1957 cross-bridge theory [31] and have
explored the chemical reactions required for myosin head attachment. Calcium ion
concentration tends to be the most common independent variable as it affects the
binding of troponin and the formation of cross-bridges.

Otazu, Futami, and Hoshimiya [41] chose to explain potentiation and the catch-
like effect based on the existence of calcium-induced calcium channels and the exis-
tence of two stable calcium concentration equilibrium points. In addition to normal
voltage-gated calcium channels, these calcium-induced channels are encouraged by
the presence of increased calcium levels to release additional calcium ions.

Likewise, Lim, Nam, and Khang modeled muscle fatigue by altering the cross-
bridge cycling velocity. Thus, reduced ATPase activity, increased H+ concentration,
and decreased intracellular pH may slow calcium reabsorption into the sarcoplasmic
reticulum. The reduced calcium levels then cause a delay in activation accordingly
[34].

Wexler, Ding, and Binder-Macleod developed a model with three coupled non-
linear differential equations. The first described calcium release and uptake by the
sarcoplasmic reticulum. The second filter modeled calcium and troponin binding and
release, and the third concerned itself with the force mechanics of cross-bridges. The
first two differential equations with respect to calcium concentration. To model force
mechanics, the third relation was based on a Hill-type model with a spring, damper, and motor in series [48]. Bobet, Gossen, and Stein did a model efficacy comparison in 2005 [5] and showed that the model by Ding et al. was one of the best at fitting isometric force. However, the parameter identification of such a procedure is nonlinear and may thus get stuck in local minima.

The drawback of biophysical models lies in their complexity and computational intractability. To address this, Zahalak and Ma attempted to describe macroscopic behavior based on microscopic mechanics using their bond distribution moment (DM) model. The bond distribution function developed by A.F. Huxley relates the fraction of cross-bridges formed to filament displacement from equilibrium (bond length). Zahalak and Ma showed that assuming cross-bridge stiffness is constant, the first three moments of the bond distribution function are proportional to muscle stiffness, muscle force, and cross-bridge elastic energy [51, 49, 50]. Thus, macroscopic outputs such as force and stiffness are explained on a molecular basis. Still, the bond distribution moment model has a large number of parameters that must be solved nonlinearly; though tractable, it lessens the DM model’s appeal.

Though physiochemical models can be good descriptors of muscle contractile dynamics, their numerical complexity makes them slow and difficult to realistically implement. Instrumentation, which may involve chemical sensors measuring calcium concentrations, is also more difficult and time-consuming.

2.3.3 Mathematical models

Muscle often exhibits non-linear behavior that is not easily described by simple physical models. This category lumps together non-physiologically based models. Some of these models are linear, while others are nonlinear. Despite muscle’s obvious non-linear behaviors, linear models have long been used to characterize muscle under certain behavioral regimes. The advantage of linear models lies in their simplicity; with them, it is easier to develop a physical intuition for predicting muscle behavior without knowing the exact model or requiring a computer to crunch through the calculations. Moreover, linear models result in speedy calculations that make the models
practical for implementation. On the other hand, nonlinear models can capture some
details in behavior that are easily overlooked by linear models. For instance, fatigue,
stretch and shortening effects, and potentiation are all nonlinear behaviors.

This section describes one popular math model, the bilinear muscle model. In the
next section, the models of interest to this thesis—the Hammerstein or Hammerstein-
Wiener nonlinear cascade with static nonlinearities acting upon dynamic linear systems—
will be reviewed in detail.

**Bilinear model**

Perhaps the simplest mathematical muscle model is the bilinear model. It uses a
least-squares fit to determine the weighting of different terms and cross terms:

\[
F = Ax + B\dot{x} + Cu + D\dot{x} + Exu + Gu + ..., \tag{2.1}
\]

where \(x\) represents length, \(\dot{x}\) represents shortening velocity, and \(u\) is the input or
activation, depending on the modeler’s choice. The cross terms enable an easy and
quick representation of activation-varying stiffness and impedance. The simplicity of
this model makes it a popular choice amongst muscle modelers.

### 2.3.4 Hammerstein model

The Hammerstein model is comprised of essentially two blocks, a static input nonlin-
earity followed by a linear dynamic system [23]:

![Hammerstein model structure](image)

Figure 2-11: Hammerstein model structure

The benefit of breaking the system model into discrete, independent blocks is
that the individual blocks may correspond to different natural phenomena whose
interactions we may understand better. The alternative to such discretized blocks is a purely computational black box approach that may work but whose inner mechanics are incomprehensible.

**History of Hammerstein cascade in muscle modeling and prostheses**

As early as 1967, scientists and engineers building prostheses began to use the Hammerstein model structure to describe muscle behavior. Vodovnik, Crochetiere, and Reswick [46] were amongst these pioneers. Their model consisted of a first-order nonlinearity that exhibited saturation and had a non-zero threshold voltage. Using this basic Hammerstein model, they developed a closed-loop controller for an elbow prosthesis.

In 1986, Hunter tested both Hammerstein and Wiener cascades in a biological setting. Whereas the Hammerstein model uses a static nonlinearity before passing through linear dynamics, the Wiener model passes the input through linear dynamics followed by a static nonlinearity. Hunter tried both models in describing muscle. However, unlike most muscle modelers, he attempted to emulate non-isometric muscle fiber, rather than whole muscle, behavior. His results showed that with the addition of the nonlinearity, both models could solve the problem of what seemed to be input amplitude-dependent time constants. He also found the Wiener model outperformed the Hammerstein one [30]. Yet, this result does not detract from the value of Hammerstein models versus Wiener since Hunter experimented with non-isometric muscle fibers rather than whole muscle. Muscle fiber is binary with either complete or no activation and therefore does not undergo muscle recruitment, the critical phenomenon that Hammerstein models seek to describe. In a separate publication the same year, he and Korenberg mentioned that the least-squares computations were more straightforward for Hammerstein than for Wiener systems [29]—another motivation for using Hammerstein models.

Also that year, Bernotas, Crago, and Chizeck developed a discrete-time Hammerstein model for isometric muscle. They tried three different discrete models and settled upon a second-order system with two poles and one zero. Varying muscle length
and stimulation frequency, the authors found that for two different muscles, soleus and plantaris, the length and frequency dependences were different: soleus acted like an overdamped second-order system at high stimulation frequencies and short muscle lengths. As the length increased and the stimulation frequency decreased, the system became less damped. Plantaris, on the other hand, was underdamped at high stimulation frequencies. This was attributed to fast versus slow muscle fiber composition. Also, the authors used recursive least squares estimation, so online identification was possible [4, 27].

Two years later, Chizeck, Crago, and Kofman used a Hammerstein model in their closed-loop pulse width-modulated controller. Their control loop was tested on two isometric cat muscles, soleus (slow-twitch) and plantaris (fast-twitch) and was robust against time-variant effects such as potentiation and fatigue. The researchers pointed out their controller design did not require a very good description of the system’s dynamics or frequency response, yet was able to perform adequately [9].

Bobet and Stein [6] investigated a variation of the Hammerstein model in 1998; their Hammerstein model was preceded by a first-order linear low-pass filter representing calcium release. The input was modulated by pulse period, which is different than most as this adds a temporal aspect to recruitment and may subsequently affect dynamics. The traditional static nonlinearity was interpreted to be saturation of calcium-troponin binding while the first-order linear output dynamics represented cross-bridge dynamics. Their model simulated both fused and unfused tetanus well but predicted isolated twitches poorly.

In 2005, Bobet, Gossen, and Stein compared simulations from seven different models to experimental data collected on isometric ankle contraction. Again, the muscles were stimulated with pulse period modulation rather than by amplitude or duration. The models, all limited to six or fewer parameters to be identified, were: (1) a critically-damped linear second-order model; (2) a general linear second-order model; (3) a third-order linear model with one zero used by Zhou et al. [52]; (4) a general linear model using Hsia’s least squares weighting function method to determine best possible impulse response assuming linearity [27]; (5) a Wiener model with
second-order dynamics and third-order polynomial output nonlinearity; (6) Bobet and Stein’s 1998 model; (7) the biophysical model of Ding et al.’s based on two differential equations for calcium activity and one for force dynamics. Bobet et al. found that the Bobet-Stein and the Ding et al. models performed best. They also stated that a critically damped second-order model was the best possible linear model, with linear models having errors of 9% or higher of maximum stimulated force. Bobet et al. concluded that a Hammerstein model in conjunction with a series spring and appropriate force-velocity relationship could yield a Hill-type model effective at simulating non-isometric as well as isometric contraction [5].

**Hammerstein model shortcomings**

The Hammerstein model has shortcomings, of course. For one, it assumes that the dynamics are linear, which is not necessarily true. It also assumes calcium dynamics are independent of recruitment. Physiologically, this is not true: Henneman’s principle dictates that small neurons innervate slow-twitch motor units, while large neurons innervate fast-twitch [25]. Thus, there is a coupling between the motor units recruited and the system dynamics. This coupling presents another system identification challenge as well: physiological recruitment begins with the finest motor units to the largest (slow to fast), while experimental recruitment, due to the crudeness of electrode stimulation, begins at the level of the largest motor units to the smallest (fast to slow). Therefore, experimental stimulation may lead to identification of a neuromuscular system that is unlike the real system in its natural working environment.

Hunt, Munih, Donaldson, and Barr [28] proposed an alternative to the Hammerstein model that addressed this coupling issue. They proposed the use of local linearized models about different activation levels and a final interpolation between local models to create a single standalone model. Gollee, Murray-Smith, and Jarvis followed up in 2001 [21] with a model which used a scheduler to weight each of the local models according to its relevance at the current operating point.

It is important to note that while such criticism is valid, the proposed alterna-
tive of multiple local linearized models is less tractable as an algorithm for practical implementation. In fact, the very same critics—Hunt et al.—chose to employ a Hammerstein model-based controller in their experimental orthotic device two years later [38].

Another pitfall of the classical Hammerstein model is its inability to describe non-isometric muscle behavior. However, the Hammerstein model may nonetheless be useful by providing an isometric model upon which a non-isometric model may be built. This approach was explored by Farahat and Herr [19]; their model combined the Hammerstein structure in series with a static output nonlinearity. The output nonlinearity has three inputs: position, velocity, and the output from the isometric Hammerstein subsystem, $z$. This is similar to a Hammerstein-Wiener cascade except that the output nonlinearity is linear in $z$; that is, $h(x, \dot{x}, z) = z \cdot h(x, \dot{x})$, where $z$, $x$, and $\dot{x}$ represent activation, muscle length, and shortening velocity, respectively.

**Convergence of Hammerstein iterative identification procedures**

In 1966, Narendra and Gallman were the first to explore an iterative procedure able to identify Hammerstein models [39]. Their strategy lay in bootstrapping: estimating one function at a time, whether it was the static nonlinearity or the linear dynamic system, then using the new estimate to determine the other function. Both the nonlinearity and dynamic difference equation were estimated by choosing linear coefficients that minimized the squared error. Narendra and Gallman did make clear that their method did not always converge to the model minimizing the mean squared error, but they showed that for some cases of polynomial nonlinearities followed by linear dynamics, their procedure could be effective and result in rapid convergence.

Stoica provided a counter-example demonstrating that the procedure was not generally convergent but commented that Narendra and Gallman’s method could still be useful if few counter-examples existed [45].

Bai and Li then proceeded to refute Stoica’s specific counter-example by illustrating how Stoica’s choice of parameter normalization resulted in unbounded errors, yet with a subtle change in normalization, the errors reach a global minimum. While
Bai and Li made no claims as to universal error convergence, they did illustrate how minor changes in iterative procedure may lead to very different convergence results, and they emphasized the utility of such an iterative identification procedure [2].

Rangan, Wolodkin, and Poola reinforced the importance of an iterative versus correlative identification method. They proved that, provided the input is white noise and the data set is sufficiently long, the iterative result will be a global minimum [42]. However, since for this thesis, there is no guarantee that our input is white, likewise, convergence is not guaranteed.

Also, though it is not applied in this thesis, using constrained rather than regular least squares methods should improve the iterative convergence of the model. By constraining the nonlinear recruitment curve to be a monotonically increasing and slowly time-varying (due to fatigue) polynomial, Chia, Chow, and Chizeck presented results which showed a recruitment curve more in line with the sigmoidal shape found in muscle literature. Their simulations with the constrained recruitment curve also showed better estimates of muscle force [11, 8].

Given the convergence issues of iterative Hammerstein identification, some have attempted to find non-iterative methods. Chang and Luus [7] did so by converting a single input-single output Hammerstein system into a multiple input-single output model. The input nonlinearity regressors constitute the multiple inputs, each filtered by the linear time-invariant dynamics. These filtered regressors are then least-squares weighted to the output. The proposed method greatly reduces computation time and can be as good an estimate as the iterative process, but it requires that disturbances be relatively low [27]. Moreover, Chang and Luus demonstrated only offline identification as their model assumed the previous outputs were all known, not estimated. Since the outputs were not rapidly changing and the true outputs were continually fed back through the system, the estimated output was not allowed to stray from the true output. Thus, it is difficult to gauge the performance of the estimated system over longer time periods, e.g., more than three time samples for a third-order difference equation.
Chapter 3

Methods and Procedures

This chapter is comprised of two main sections: experimental procedure and simulation methods. Experimental procedure includes hardware setup and experimental protocol; it begins this chapter.

3.1 Hardware

3.1.1 Experimental set-up

The non-biological hardware was the same as that from Farahat and Herr [18] and consisted of eight fundamental components:

- two computers, one of them the user interface and the second a machine that did all of the processing;

- a voice coil motor which enforced a muscle length boundary condition-in this case, a constant length;

- a load cell which measured muscle force output;

- a linear encoder which measured the position of one end of the muscle and whose reading was fed back to maintain the isometric boundary condition;

- a suction electrode comprised of a syringe entwined with fine silver electrode wire that directly contacted the sciatic nerve of the muscle. The syringe tip's
pressure difference helped maintain electrode contact by sucking the nerve end close to the electrode wire;

- an electrical power supply providing up to at least 18V;

- a circuit board specially built and programmed to deliver bipolar stimulator pulses between 0 to 18V with 1\(\mu\)s timing resolution.

The circuit board was connected to a data acquisition board in a specially dedicated computer; this machine was booted to exclusively run the MATLAB xPC Target kernel and handled all the data acquisition as well as commands to the stimulator board. The user issued high-level commands via a GUI on the xPC host machine. These commands were then communicated to the target machine through a TCP/IP connection. Lower-level functions were directly programmed into a microcontroller on the circuit board.

Experiments were run by an identification experiment model built in the MATLAB Simulink xPC Target environment. The Simulink model dictated stimulation
parameters and onset of stimulation while recording muscle output force and length via a load cell in series with the muscle and via a linear encoder, respectively. Time and the varying stimulation parameter were also recorded.

3.1.2 Muscle set-up

The experiments were performed on leopard frog (*Rana pipiens*) plantaris longus muscles. To extract the muscles, the frogs were anesthetized in icy water, then euthanized via a double pithing procedure (severance of the brain and spinal cord). Next, the skin was removed from the thigh and calf. Then the thigh was positioned at a ninety-degree angle from the body, and the calf was positioned perpendicular to the thigh. Silk suture was tied around each end of the plantaris longus muscle as close as possible to the hip and knee joints. The distance between sutures was measured in vivo and recorded as the rest length of the muscle. Although the suture distance was not equivalent to muscle belly length since tendon length was inevitably included in the measurement, tendon has experimentally been shown to be much stiffer than muscle [3]. Therefore, any changes in suture distance were practically equivalent to changes in muscle belly length. Moreover, suture distance was readily identifiable whereas the muscle belly length was not quite as apparent since the muscle and tendon fused into one another. Having a known rest length was important because of muscle's force-length dependence and because maximal force was produced at rest length when cross-bridge formation could be maximized.

After measuring the suture distance, the muscles were cut out of the body with small bone chips attached to either end. These chips helped prevent suture slippage. A length of the sciatic nerve attached to the muscle was also preserved. The sutures were tied to small dove-tailed acrylic mounts, and the mounts were affixed, the knee mount to the load cell and the ankle mount to the voice coil motor (VCM). The VCM was then calibrated so its reference position held the muscle at rest length. Last, the suction electrode was positioned in direct electrical contact with the sciatic nerve but not the rest of the muscle; instead of applying charge directly to the muscle tissue, this emulated natural recruitment as much as possible.
Throughout extraction and experimentation, the muscles were kept moist with commercial amphibian Ringer's solution (Post Apple Scientific, Inc.). Please see Figure 3-1 for a diagram of the experimental hardware.

3.2 Experimental procedure

One frog provided two plantaris longus muscles. Experiments were conducted on one muscle at a time: while the first muscle underwent trials, the second muscle was refrigerated to slow its metabolism and thus minimize necrosis of the core tissue. Fourteen to fifteen tests were performed on each muscle.

Each test was conducted in three stages. The first stage measured the current position of the linear encoder. The second stage compared the current and reference positions, then commanded the voice coil motor to make a smooth, linear transition to the desired position. The third stage used the voice coil motor to maintain the reference position during isometric stimulation of the muscle. Note that for higher muscle force output, some muscle shortening occurred due to VCM compliance.

The stimulation input consisted of a bipolar pulse train with varying pulse width; amplitude and pulse count remained constant. The choice of pulse width, rather than amplitude, as the varying parameter originated from literature demonstrating that pulse width modulated force just as well as pulse amplitude [9] and, furthermore, was less damaging to tissue [10, 37]. Bipolar pulses were recommended as they lessened tissue damage as well as prevented electrode corrosion [33, 13].

Since the Simulink model recorded actual stimulator triggers (which could include random pulse period variation, or “jitter”) rather than pre-programmed pulse periods (no jitter), pulse period was not necessarily fixed. Pulse period jitter was added as a possible method of enhancing the identification process as it could cover a slightly wider range of dynamic response in a single experiment. Moreover, pulse period jitter could have helped prevent muscles from fatiguing due to very regular, cyclic stimulation. Owing to the stimulator cards’ design and programming, pulse period was interpreted as the time between the end of one pulse and the onset of a second
rather than the traditional definition as the time from the onset of one pulse to the onset of a second pulse.

Experiments lasted seven seconds: three to allow the VCM to servo to its reference isometric position and four for actual muscle stimulation (unless otherwise noted in Appendix A). Tests were conducted with 10V-pulses at frequencies between 1Hz to 100Hz. Each stimulation consisted of only one pulse to avoid modulation simultaneously by pulse width and pulse period as demonstrated by [10, 6] and to avoid temporal effects on dynamics. A few tests were conducted with frequency jitter in which pulses could be delayed up to 90% of the stimulation period minus maximal pulse width. Pulse width varied between 0 to 1ms with a resolution of 1μs.

For further specifics on each experimental run, Appendix C includes notes from each test.

3.3 System identification and simulation

The bulk of this thesis’ work was on system identification of isometric muscle. Once the experiments were completed, curve-fitting was conducted on data from the one muscle that underwent different stimulation frequencies. Each data set was divided into two sections: a training portion to estimate the static nonlinearity as well as the linear dynamic system (recall Figure 4-1) and a validation portion which used the same static nonlinear function and linear dynamic system but with a different input.

The input was the electrical pulse train delivered to the muscle via suction electrodes in contact with the sciatic nerve. In our model, this signal was then sent through the input nonlinearity block, which may be interpreted as an isometric recruitment curve (IRC) [15]. The output signal was filtered by a dynamic system. Employing the common simplification that the muscle was purely isometric, the output of the dynamic filter was equivalent to force produced by the muscle.

Each data set was identified in two parts. The first involved training the model to determine what coefficients and system orders were appropriate. The second half involved inputting the experimental stimulation into the model trained by the first
half. This would produce a predicted output force. For the figures in this thesis, blue lines plot the measured force, while both red and green lines plot the estimated force. The red lines illustrate the interval over which training of the model occurred.

Identifying each data set’s Hammerstein system involved two main challenges: determining the static input nonlinearity and the linear dynamics.

3.3.1 The static input nonlinearity

The static input nonlinearity was estimated using linear combinations of third- to fifth-order Chebyshev polynomials. The coefficients of each of the Chebyshev polynomials were determined using a least-squares regression. There are several ways to do least squares regression, but a linear (in the coefficients) least-squares is the simplest. It is known as the Moore-Penrose pseudoinverse.

Use of the Moore-Penrose pseudoinverse

For systems $Ax = b$ where the $A$-matrix is not square or full rank or for systems where $Ax$ only approximates $b$ so there is no exact solution, the regular inverse $A^{-1}$ cannot be found. The pseudoinverse is calculated instead. In curve fitting—an instance of $Ax \approx b$—an effort is made to determine the best $x$ such that the error $Ax - b$ is minimized. However, instead of using a direct measure of error, which may be positive or negative and thus result in some cancellation, the square of the Euclidean norm (“length”) of the error is minimized:

$$\frac{d}{dx} ||Ax - b||^2 = \frac{d}{dx} ((Ax - b)^T (Ax - b))$$

$$= \frac{d}{dx} ((Ax)^T Ax - (Ax)^T b - b^T (Ax) - b'^T b)$$

$$= \frac{d}{dx} ((Ax)^T Ax - 2(A^Tx)b - b^T b)$$

$$= 2A^TAx - 2A^Tb = 0$$

This leads to the solution $x = (A^TA)^{-1}A^Tb$, where $(A^TA)^{-1}A^T$ is the pseudoinverse.

In the curve-fitting procedures for this thesis, the pseudoinverse was used to de-
termine coefficients that weight components depending on their contribution to the final output. In this case, the output was force. Note that this iterative procedure remained a linear least squares fit: it was linear in the coefficients though the basis functions themselves were nonlinear.

**Basis functions for the input nonlinearity**

Despite physiological evidence that supports a sigmoidal-shaped isometric recruitment relationship between input parameter value (one of a few different things: pulse amplitude, pulse width, pulse count, or pulse frequency) and activation, the basis functions used in this thesis were not sigmoids. Sigmoid functions may be represented by a function of the form:

\[ S = \frac{1}{1 - \exp(-\alpha(x - x_0))}, \]

where \( \alpha \) changes the slope of the sigmoid and \( x_0 \) changes the point at which the sigmoid passes through 0.5. To change the shape of this sigmoid would require a nonlinear regression in \( \alpha \) and \( x_0 \), and of course, a nonlinear regression involves more difficult computations than a linear regression. Thus, instead of attempting to directly fit a sigmoid, two other types of basis functions were tried: radial basis functions and Chebyshev type I unshifted polynomials.

**Radial basis functions**

Radial basis functions (RBFs), which are essentially Gaussians with specified variances and centers, are described mathematically by:

\[ A(u) = \exp \left( -\frac{|u - \zeta|^2}{\sigma^2} \right), \]

where \( \zeta \) and \( \sigma \) are the centers and spreads of the Gaussians. They are often used because they are simple to implement in multiple dimensions, and the variances and centers can be tweaked to be nearly, but not completely, orthogonal and independent of each other. Moreover, it is relatively simple to get an intuition for how all the weighted RBFs can be summed to resolve a curve.
Chebyshev polynomials

Chebyshev polynomials are also popular for curve-fitting procedures and were also tried during the curve-fitting procedures. Chebyshev polynomials obey the following orthogonality condition on the [-1,1] interval:

\[
\int_{-1}^{1} T_m(x) \cdot T_n(x) \cdot \frac{1}{\sqrt{1-x^2}} = \begin{cases} 
0 & \text{if } m \neq n, \\
\pi & \text{if } m = n = 0, \\
\frac{\pi}{2} & \text{if } m = n \neq 0.
\end{cases} 
\]  

(3.4)

Orthogonal polynomials are desirable in curve-fitting for a number of reasons; most importantly, they run into fewer problems when calculating the matrix’s inverse. The factor \(\frac{1}{\sqrt{1-x^2}}\) implies that Chebyshev polynomials are only orthogonal on the [-1,1] interval for a given input probability distribution. Experimentally, this is impossible to achieve: there is never infinite probability at the extreme ranges of the data set. However, though Chebyshev polynomial regressors will not be completely orthogonal, they have similar variances (their amplitudes on the [-1,1] interval are roughly the same), so a regression matrix composed of Chebyshev polynomials should be relatively well-conditioned when taking the pseudoinverse (Westwick and Kearney [47, pp 29-33]). In addition, because they are roughly orthogonal, Chebyshev polynomials may approximate a curve more concisely using fewer numbers of polynomials.

There are several types of Chebyshev polynomials. Those of the first kind, \(T(x)\), are more frequently used than those of the second, \(U(x)\). The two kinds have different input probability distributions: \(T(x)\) uses \(\frac{1}{\sqrt{(1-x)^2}}\), while \(U(x)\) uses \(\sqrt{(1-x)^2}\). Both kinds may also be shifted to be orthogonal on the [0,1] interval. In this thesis, unshifted Chebyshev polynomials of the first kind were used. They are generated by
the recurrence relation:

\[
T_0(x) = 1, \\
T_1(x) = x, \\
T_{n+1}(x) = 2x \cdot T_n(x) - T_{n-1}(x). 
\] (3.5)

When constructing the regression matrix, each column was composed of one Chebyshev polynomial whose input \( x \) was merely the input to the system at that time sample. The length of each column was equivalent to the length of the data set.

### 3.3.2 The linear dynamic system

The second challenge in Hammerstein system identification was determining the linear dynamics best describing each data set. The linear dynamic system was estimated using various functions from the MATLAB System Identification toolbox.

The most important modeling choice was the disturbance model selection. Several disturbance models are available from the System ID toolbox, such as ARX (autoregressive with exogeneous input), ARMAX (autoregressive with moving average exogeneous input), and OE (output error) models. The output error model was chosen on the basis of noise characteristics; it assumes noise is added directly to the measurement of output force and does not enter into any of the system dynamics. A mathematical description of the discrete-time model is:

\[
y(t) = \frac{B(q)}{F(q)} \cdot u(t - nk) + e(t), 
\] (3.6)

where \( nk \) is the delay and \( q \) is the shift operator (commonly also denoted as \( Z^{-1} \) in the signal processing field or \( Z \) by controls engineers). Thus, \( \frac{B(q)}{F(q)} \) represents a discrete-time difference equation. The orders of \( B(q) \) and \( F(q) \) are specified by the user; if the user dictates that both orders be equivalent, MATLAB fits a \( F(q) \) of the specified order and generally a \( B(q) \) of one order less than specified. Using the output force data and the guessed \( v = f(u) \), a dynamic model—linear in its coefficients—was
estimated by the oe command.

**MATLAB's oe command**

A crucial part of the identification process occurs within MATLAB's oe command, so the command warrants some discussion. The oe code finds parameter coefficients using the prediction error method in which the minimized criterion is the square of the filtered error, normalized by the length of the data set. The linear filter is created by multiplying the input frequency spectrum by the inverse noise model and is referred to by MATLAB as prediction weighing. This filter essentially weights the transfer function fits more heavily for frequencies containing more spectral power.

\[
\hat{\theta}_N = \arg\min_{\theta \in \mathcal{D}_\mathcal{M}} \frac{1}{N} \sum_{t=1}^{N} \frac{1}{2} (L(q) \cdot \epsilon(t, \theta))^2
\]

(3.7)

In the above equation, \( \hat{\theta}_N \) refers to the parameter estimates (e.g., difference equation coefficients), \( N \) to the length of the data set, \( \mathcal{D}_\mathcal{M} \) to the set of models to which the estimated model belongs, \( L(q) \) to the prediction weighing filter mentioned above, and \( \epsilon \) to the errors between estimates and actual data. Note that this is a nonlinear optimization as \( \epsilon \) is both a function of time \( t \) and parameter estimate \( \theta \).

There are several algorithm options available for oe. The algorithm itself is iterative, and by default, the maximum number of iterations is twenty. For this thesis, the only additional specified property was that the focus be on stability. MATLAB's estimation of a stable system uses the prediction weighing method mentioned above.


**Simulation initial conditions**

Once the coefficients of the discrete transfer function \( \frac{B(q)}{F(q)} \) were found, the transfer function system representation was converted to a state-space representation. State-space was preferred over a transfer function representation as it allowed specification.
of non-zero initial states for the validation half of the data. The initial conditions for the training half of the data were zero. For a discrete system, the states could be previous values of the output. However, in simulation, when the states and the force estimate were plotted, the states were indeed previous force estimates, but they were also multiplied by a gain. This gain may be explained by examining the generic continuous-time state-space representation:

\[
\dot{x}(t) = Ax(t) + Bu(t) \\
y(t) = Cx(t) + Du(t - t_{\text{delay}}) + e(t)
\] (3.8)

The value of gain observed is the reciprocal of the coefficient \( C \). Thus, the non-zero initial states of the validation data were more easily calculated using a state space representation.

### 3.3.3 Bootstrapping

Before the iterative process began, a number of critical modeling decisions must be made. What order should the dynamic system be? What should be the largest order Chebyshev polynomial to be included in the nonlinear fit? Muscle is sometimes described as a critically-damped second-order dynamic system, but higher order behavior would be neglected by such a simplistic model. It was unclear what order the model and the nonlinear fits should be, so combinations of dynamic systems between second- to fourth-order and Chebyshev polynomials between first- to third-order were tested on sixteen data sets. Purely dynamic systems between second- and fourth-order were also simulated. Root mean-squared errors were recorded for each simulation, and based on the errors and quality of fits, the orders of the dynamics and the polynomial nonlinearity were chosen.

Hammerstein model structures require two functions: a static input nonlinearity and a linear dynamic system. When iterating to estimate the model, either the static nonlinearity or the linear dynamics may first be estimated. In this case, the initial estimate was of the static nonlinearity; it merely guessed \( v(t) = f(u) = u \).
Having assumed a $v(t)$, the linear dynamic system was then estimated between $v(t)$ and $f(t)$ using MATLAB's `oe` command to create an output error model. The algorithm was implemented with an emphasis on identifying stable systems, but stability was not guaranteed.

Finding the nonlinearity composed of weighted Chebyshev polynomials was the next step. Calculating the inverse dynamic system, estimating the new $v(t)$ based on $f(t)$, and then finding the polynomial coefficients would require three steps. Additionally, it would require fitting polynomial weighting coefficients to a $v(t)$ that would be heavily reliant on the estimated dynamic system. Instead, system linearity was exploited. It presented a less computationally expensive alternative that fit the regression matrix directly to measured output $y(t)$ rather than an estimated $v(t)$. In other words, it was assumed that the model was still linear: a gain at the beginning of the system was equivalent to a model with the same gain being multiplied once anywhere else in the system. In this case, the regression matrix was passed through the newly-estimated dynamic system, and a least-squares fit was then conducted directly between that filtered matrix and the measured output force. This produced an estimate of the regressor coefficients, which were then weighted ($\alpha = 0.5$) and added to weighted ($\alpha = 0.5$) previous regressor coefficients. $\alpha$ could be changed between 0 (use only new coefficients) and 1 (use only old coefficients), and was enacted to lessen rapid fluctuation of the input nonlinearity. On the down side, it slowed down model convergence. The matrix product of the regression matrix and the weighted-sum coefficients constituted the new estimate of $v(t)$.

The process then repeated by re-estimating the dynamic system based on the new $v(t)$ and the measured force $f(t)$. The iterative procedure continued until the error metric reached a pre-specified threshold. Here, the stopping criterion required that the standard deviation of the last five errors, normalized by the last error, be less than five percent. Stability of the converged estimate was also recorded: if the H2 norm of each dynamic estimate was not infinite, the estimate was stable.

As a cautionary note, this iterative procedure is not guaranteed to find a global minimum. Narendra and Gallman [39] employed an iterative process to identify
known Hammerstein systems, and they showed it worked for their four test cases. A counter-example was provided in 1981 [45], but the same was refuted in 1995 by Rangan et al. [42]. Additionally, Rangan et al. claimed that convergence was guaranteed provided the data set was sufficiently long and the input was white. In summary, error convergence of iterative Hammerstein identification remains debatable, and since we had no guarantee that our input was white, neither could we ensure simulation convergence.

As standard practice, the low frequency gain of either the nonlinearity or the dynamic system is normalized. The gain is preserved by multiplying it with the other function. Here, the recruitment curve was normalized to unity to represent full activation.

### 3.3.4 Simulation order

With several available parameters with which to play, the first question became, what orders of dynamic system and polynomial nonlinearity were appropriate? The first batch of simulations were conducted on second-, third-, and fourth-order dynamic systems without any input nonlinearity. From the results, it became easier to justify the choice or elimination of a particular dynamic system order. The next step was to add the nonlinearity to the choice(s) of dynamic systems. These simulations were run with first-, second-, and third-order polynomial nonlinearities. Finally, simulations were run with fixed nonlinearities and/or dynamics to determine how well one model, trained by one data set, could estimate the output of another data set.
Chapter 4

Results and Discussion

This chapter presents results and discusses them immediately. The first section uses a prototypical example to review the identification process. The ensuing sections examine questions presented by the identification process; in order to determine the effectiveness of Hammerstein models across stimulation frequencies, several additional questions must be asked. For instance, is the input range adequate to cover the force behaviors in which we are interested? What order of linear dynamics is appropriate? What should the highest-order polynomial regressor be for the input recruitment nonlinearity? Can fixing the nonlinearity produce better curve fits when the data is unable to inform the recruitment curve? Such are the questions asked and answered in this chapter. By incrementing the modeling degrees of freedom, it is possible to see which, if any, are drastically beneficial to the curve-fitting process. The best fits that the algorithm can produce may then be compared across input frequencies to determine up to what frequency the models generate reasonable estimates.

Note that all plots, unless otherwise noted, show only results from stable estimated systems. Logs of experimental and simulation notes are contained in Appendices A and C. Meanwhile, specific examples and figures are contained in this chapter to illustrate particular points of discussion.
4.1 A prototypical system identification

To review, the Hammerstein model structure is shown below. The notation, as introduced in §2.3.4 is as follows: $u$ is the system input to the static nonlinearity, $v$ is the output of the nonlinearity and input to the dynamics, $z$ is the measured system output, and $\hat{z}$ is the estimated output from the dynamics. Physiologically, these variables may be interpreted as the input pulse train, the activation level, the measured contractile force, and the estimated force, respectively.

The following sample identification process was conducted with third-order dynamics, third-order Chebyshev polynomial regressors, and a pure delay of 2.5ms. The first step in the identification process was making an initial guess of the nonlinearity or dynamics. For this thesis, the initial guess of the nonlinearity was $v = f(u) = u$—in other words, a linear recruitment curve with no gain. The algorithm began by making an estimate of the dynamics between $v = u$ and $z$ using the oe command. The command outputted a discrete-time filter through which the Chebyshev polynomial regressors of $v$ are passed. A least-squares fit was made between the regressors and the measured data, and those coefficients dictated the new $v$. The new $v$ was then passed through the filter to create an estimated force $\hat{z}$. This completed the first iteration; the resulting recruitment nonlinearity and estimated force are shown in Figure 4-2(a).

The second iteration then used the newly-estimated $v$ and measured force $z$ to get a new estimate of the dynamics, and the process began anew. After the second iteration, the recruitment nonlinearity and estimated force were as shown in Figure 4-2(b).

Figure 4-3 shows the recruitment nonlinearities and estimated forces after three,
(a) Force and isometric recruitment curve after one iteration

(b) Force and isometric recruitment curve after two iterations

Figure 4-2: Example identification after 1 and 2 iterations (2Hz data)
Figure 4-3: Example identification after three, four, and twelve iterations (2Hz data)
four, and twelve iterations. By the twelfth iteration, the algorithm converged; that is, the standard deviation of the last five RMS error values reached less than 5% of the last RMS error value.

This identification procedure included a number of parameters which were specified by the user, such as input range, dynamic system order, the existence and duration of pure delays, the choice of regressor basis functions, and so forth. In order to determine what each of these parameters should be, studies were conducted beforehand to explore the options for each of these parameters. The ensuing sections detail the investigations into each of these questions. We begin with the question of whether or not force saturation was being reached with the input range used.

4.2 Is the input pulse-width range adequate to reach force saturation?

One of the questions that arose was whether or not the range of pulse widths was adequate to reach saturation of the contractile force and whether or not the polynomial estimates of the recruitment curves truly reflected the recruitment behavior of the muscles. Assuming the dynamics were purely linear, each pulse should elicit a force peak of proportional magnitude.

These questions were addressed by plotting the magnitude of the individual force peaks versus the inputted pulse widths for the corresponding twitches. The force peaks were obtained after low pass filtering the raw data using a second-order Butterworth low pass filter with a cutoff frequency of 100Hz, then differentiating and looking for zero crossings. Since each twitch must be isolated in order that the dynamics are not affected by a previous input, the following exercise was conducted primarily for low stimulation frequency data sets. At high frequencies, it was more difficult to ascertain the discrete peaks. Figures 4-4(a) and 4-5(a) illustrate such a relationship between force and pulse width for two stimulation frequencies, 1 and 2Hz. The relationships were obtained in a manner similar to Durfee’s peak impulse
method [15] (see §2.2.2).

Figure 4-4: Isometric recruitment relationship - 1Hz

Figure 4-5: Isometric recruitment relationship - 2Hz

Figures 4-4(a) and 4-5(a) demonstrate that increasing pulse width did indeed result in recruitment of motor units to produce higher force levels. All three plots also confirm the pulse width range between 0.1 to 1ms was adequate to attain force saturation.
4.3 Can the algorithm identify the appropriate recruitment curve?

Figures 4-4(b) and 4-5(b) are third-order polynomials obtained, in concert with third-order dynamic systems, by the iterative identification algorithm used by Hunter and Korenberg [29] and mentioned by Durfee and MacLean [15]. The figures show the algorithm produces recruitment nonlinearities resembling those found by the peak impulse method (see Figures 4-4(a) and 4-5(a)). In addition, they are consistent with the recruitment curves found in the literature.

4.4 What basis functions should be used to fit the input nonlinearity?

One of the crucial decisions was what basis function to use as a regressor for the static nonlinearity. Three types of basis functions could potentially be tried: sigmoids, radial basis functions, and Chebyshev polynomials. Sigmoids, though similar in shape to experimental recruitment curves, were scrapped because they require nonlinear regressions rather than straightforward least-squares estimates. Likewise, radial basis functions (RBFs) often produced very wavy recruitment curves with widely fluctuating coefficients. This was especially true if the variances were relatively small and the centers not particularly numerous. Such fluctuations are indicative of a large condition number—meaning, the coefficients have great relative sensitivity to noise (Westwick and Kearney [47, p 29]). Intuitively, this may be explained by the lack of data points near a particular RBF center. Thus, the uninformed RBF center may be heavily weighted; instead of considering its weighting coefficient based on contribution of the peak to the overall curve, the pseudoinverse determines the weight based on contribution of the RBF’s low amplitude edges to the overall curve. These weights often lead to drastically inaccurate curve fits. Due to the waviness and unreliability of using RBFs for fitting, they were scrapped in favor of Chebyshev polynomials, which
were then used across all the simulations described below.

4.5 What is the appropriate input delay?

Delay selection was important as the algorithm often estimated non-minimum phase responses when given inappropriate delays, as seen in Figures 4-6 and 4-8 (blue = measured force, red = estimated force over training interval, green = estimated force over validation interval, - = onset of stimulation). To determine what number of delays would suit best, the delay parameter of Matlab’s oe command was increased from between one to thirty-five delays for 3, 5, 7, 10, 20, 75, and 100Hz data. The algorithm was then run with a second-order polynomial nonlinearity and third-order dynamics. The delay that generated least simulation error was noted for each frequency. One and five delays seemed to produce lower errors. With the data sampled at 2kHz, five delays (2.5ms) seemed like a more reasonable delay than 1 delay (0.5ms). Physiologically, such a time scale is reasonable since the chemical processes involved in muscle contraction—transmission across the motor end plate, muscle fiber conduction, and excitation-contraction coupling [52]—take a little time. As an example, Wexler, Ding, and Binder-Macleod chose 4ms as the refractory period of calcium ion channels [48]. Hill concluded that frog muscle at 0°C takes less than 20ms to shorten (McMahon [36, pp 17-19]); however, the muscle used in this thesis’ experiments were stimulated at room temperature, so one would expect a shorter delay.

<table>
<thead>
<tr>
<th>Frequency (Hz)</th>
<th>Delay for least RMS error</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>5</td>
<td>15</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
</tr>
<tr>
<td>10</td>
<td>1</td>
</tr>
<tr>
<td>20</td>
<td>5</td>
</tr>
<tr>
<td>75</td>
<td>1</td>
</tr>
<tr>
<td>100</td>
<td>5</td>
</tr>
</tbody>
</table>
Figure 4-6: Delay selection: 3Hz data
Figure 4-7: Delay selection: 7Hz data (continued on next page)
Figure 4-8: Delay selection: 7Hz data (continued from previous page)
4.6 How does a dynamic system with no input nonlinearity perform?

The next step was to test how well a simple, linear time-invariant dynamic system with no input nonlinearity would perform. The dynamic estimates were obtained using Matlab’s oe command for output error model identification. Second-, third-, and fourth-order models were applied. Figures 4-9, 4-10, and 4-11 illustrate simple dynamic fits on three data sets in which muscle was stimulated at 3, 40, and 100Hz. Experimental force data is shown in blue, estimated force in red, and asterisks mark the onset of stimulation pulses.

For isolated muscle twitch response at 3Hz stimulation, the second- and fourth-order estimates resulted in non-minimum phase zeros. The fourth-order estimate also oscillated rather than returned to its zero-force rest state.

For unfused tetanic muscle response due to 40Hz stimulation, second-order dynamics tended to follow low frequency behavior less well ($e_{RMS} = 10.9\%$ of $F_{max}$) than third- or fourth-order dynamics ($e_{RMS} = 8.5\%$ and $6.9\%$, respectively).

For fused tetanus due to 100Hz stimulation, all fits failed. Only the fourth-order estimate slightly resembled the measured force, but all of the estimated systems were unstable. Behavior due to 100Hz stimulation was nonlinear; with nearly continual stimulation, one would expect a step response from the muscle, but this is not what happened. Instead, the muscle response violated one of the Hammerstein model tenets—that the dynamics are linear. If they were linear, the force should be sustained. Realistically, muscle fatigues during prolonged tetanus, thus introducing a nonlinearity that the identification algorithm is, at present, incapable of estimating. In order to model fatigue, a time-dependent nonlinearity would need to be added to the Hammerstein cascade. That the dynamics were not even able to construct the first-order step response may be attributed to the lack of information content when identifying tetanic contraction: the algorithm observed that regardless of the level of input, the force response was nearly constant. As a result, noise behavior may easily have dominated the identification algorithm, and unsurprisingly, the resulting
Figure 4-9: Simulations with no nonlinearity: muscle response at 3Hz stimulation
Figure 4-10: Simulations with no nonlinearity: muscle response at 40Hz stimulation
Figure 4-11: Simulations with no nonlinearity: muscle response at 100Hz stimulation (all have unstable dynamics)
estimates were very poor.

![Figure 4-12: Simulation errors with no nonlinearity](image)

Errors based on linear dynamics-only curve fits (shown in Figure 4-12) helped determine what dynamic system would be most appropriate for the Hammerstein muscle model. The variation of difference equation coefficients (see Figure 4-13) also informed the choice of dynamic system order.

Although fourth-order models generated the least error, particularly at high frequencies, it was not apparent that the physical system was predominantly fourth-order or that the identification algorithm was capable of identifying a consistent fourth-order model for the data. The relatively large amount of difference equation coefficient variation for fourth-order models (average standard deviation $\sigma = 1.3$ over four coefficients versus $\sigma = 0.0095$ and $0.027$ for second- and third-order models, respectively) seemed to suggest the algorithm was capable of finding many different solutions to fit the data but which did not necessarily uniquely describe the system. Thus, a decision was made to proceed simulating with only second- and third-order dynamic models.
Parameter variation: 2nd order dynamics, no nonlinearity

(a) second-order dynamics with no nonlinearity

Parameter variation: 3rd order dynamics, no nonlinearity

(b) third-order dynamics with no nonlinearity

Parameter variation: 4th order dynamics, no nonlinearity

(c) fourth-order dynamics with no nonlinearity

Figure 4-13: Difference equation denominator coefficient variation
4.7 How do different combinations of input polynomial nonlinearities and dynamic systems perform?

Since it remained unclear what combination, if any, of polynomial nonlinearities and linear dynamics would consistently produce the best estimates of force, all combinations of linear second- and third-order dynamic systems with first- through third-order polynomial nonlinearities were simulated. Figure 4-14 plots, as scatter and bar graphs, the simulation errors across all stimulated frequencies for one muscle.

Figure 4-14 shows that at frequencies below 20Hz, linear and nonlinear models produced simulations errors that were relatively comparable ($e_{RMS} = 12.4\%$ and $12.3\%$, respectively) and dynamically stable ($26/27 = 96\%$ of linear simulations and $52/54 = 96\%$ of nonlinear simulations). For frequencies between 20 to 40Hz, the addition of a nonlinearity resulted in errors ($\bar{e}_{RMS} = 10.5\%$ of $F_{max}$) comparable to those of linear models ($\bar{e}_{RMS} = 10.2\%$ of $F_{max}$), but algorithm convergence and dynamic stability was not as reliable ($6/18 = 33\%$ of nonlinear simulations versus $8/9 = 90\%$ of linear simulations). For frequencies above 40Hz, neither the linear nor nonlinear models fared well; dynamic estimates were often unstable or the algorithm did not converge ($2/9 = 22\%$ of linear simulations, $8/18 = 44\%$ of nonlinear simulations).

It should be noted that dynamics order selection based purely on errors would be naive. Based on strictly RMS errors, it would seem that second-order dynamics sometimes describe the system better than third-order. However, third-order models should not be discounted; as Figure 4-15 reveals, there can be discrepancies in which the dynamic system producing lower errors has a poorer qualitative fit. In this case, the second-order system generates lower errors, but has larger non-minimum phase dips.

Both second- and third-order dynamics are reasonable. Though many muscle models assume critically-damped second-order dynamics, a third-order model is not highly unusual. Ettema and Meijer used one in their exponential decay model [17].
Comparison of different polynomial nonlinearities with second- and third-order dynamics

Figure 4-14: Simulation errors of different dynamic systems with different nonlinearities (in scatter and bar graph format)
Zhou et al. chose to describe their muscle as a second-order system, but the system became third-order when modeling the muscle-joint in order to take into account the viscoelastic properties of tendon and ligament [52]. Meanwhile, Donaldson et al. chose a third-order linear transfer function to model isometric force due to a pulse train with constant amplitude and pulse width but random stimulation periods between 1 to 70ms [12]. In general, when researchers have used third-order relationships, the transfer functions have been between output force and position for their Hill-type mass-spring-damper models; however, in our case, the third-order dynamics are between output force and a voltage-based pulse train. Physiologically, voltage drives the release of ions and incites a diffusion-based reaction. Non-steady state diffusion is a second-order phenomenon, as described by Fick’s second law:

\[
\frac{du}{dt} = k \frac{\partial^2 u}{\partial x^2},
\]

where \( u \) may be ion concentration, \( k \) is a diffusivity constant, and \( x \) represents the direction(s) of diffusion [22]. A second-order diffusion-based system may be cascaded with a first-order relationship similar to the force generator model proposed by Wexler, Ding, and Binder-Macleod, who modeled the force mechanics as a motor with a linear force-velocity curve and who related shortening velocity with calcium concentration [48].
4.8 How does a system with fixed input nonlinearity perform?

Given that muscle modelers already have an idea what the recruitment nonlinearity should be, some of the modeling degrees of freedom could be removed by fixing the nonlinearity. Figures 4-4 and 4-5 demonstrated that the iterative identification algorithm was capable of finding the recruitment nonlinearity, so the first step was to determine what recruitment curves were generated across all data sets, then to pick a suitable one based on the widely-accepted view of recruitment as a sigmoidal curve. The selected nonlinearity would be used as the recruitment curve for all data sets to further obtain estimates of the dynamics.

Most muscle models in the literature create the recruitment as either a piece-wise linear or a polynomial-composed sigmoidal curve. Of the third-order polynomial recruitment curves obtained in conjunction with third-order dynamic estimates (shown in Figures C-8 and C-9), the one from experiment 15 at 2Hz was most sigmoid-like and resembled recruitment curves discussed in the literature. The polynomial regressor coefficients obtained from that fit were then applied across all data sets to fix the nonlinearity while allowing the algorithm to find dynamic estimates tailored for each data set. The resulting curve fits may be seen in Figures C-10, C-11, and C-12. A comparison of simulation errors for third-order dynamics with both fixed and iteratively-found third-order polynomial input nonlinearities may be seen in Figure 4-16, and Figure 4-17 shows difference equation denominator coefficient variation.

Figure 4-16 suggests that at frequencies of 20Hz or lower, fixing the recruitment nonlinearity produced comparable results ($\bar{e}_{RMS} = 16.3\%$ over 12 simulations) as iteratively finding both the nonlinearity and dynamics ($\bar{e}_{RMS} = 16.6\%$ over 66 simulations). Figure 4-17 shows how little variation in difference equation coefficients there is across data sets identified with a fixed nonlinearity ($\bar{\sigma} = 0.020$); this demonstrates that the estimated dynamics were consistent across simulations and were therefore relatively reliable. This is a useful conclusion, as the estimation process may be reduced from a potentially time-consuming bootstrapping procedure to one step: identifica-
Benefit of iterative procedure versus fixing nonlinearity

Figure 4-16: Simulation errors for fixed and iteratively-found recruitment nonlinearity (third-order dynamics, third-order polynomial)

Figure 4-17: Difference equation denominator coefficient variation
tion of the dynamics. At stimulation frequencies above 40Hz, fixing the recruitment nonlinearity did not result in better dynamic stability or algorithm convergence (one stable estimate with $\varepsilon_{RMS} = 53.6\%$ of the maximum force for that data set). One would expect the simulations with the fixed nonlinearity to be somewhat better since the algorithm only needs to concentrate on dynamics. However, as in the other identification attempts, the estimated force often does not follow the measured behavior at all. This was probably because the information content of the data was low, so the resulting lousy dynamic estimate was dominated by noise behavior. Potentially, the fixed nonlinearity may have been inappropriate at higher stimulation frequencies, but this remains unclear.

4.9 How does a system with fixed input nonlinearity and fixed dynamics perform for high stimulation frequency input?

Muscle's force response due to high frequency stimulation demonstrates two dominating nonlinear behaviors: force potentiation and fatigue.

Since simulation of high frequency stimulation input with the fixed nonlinearity still produced poor results, the question was brought up that perhaps the data contained too little information and too much noise to estimate the dynamics adequately. In that case, it was interesting to see if applying the recruitment nonlinearity and dynamics found from a data set with well-defined twitches to a data set with high frequency stimulation input would be effective in describing muscle behavior under high frequency input. The nonlinearity regressor coefficients and third-order difference equation coefficients from 2Hz data (Experiment ID.015) was applied to 50, 75, and 100Hz data, and the resulting simulated responses are shown in Figure 4-18.

While the estimates showed improvement ($\bar{\varepsilon}_{RMS} = 65.7\%$ of $F_{\max}$) over those found with linear models ($\bar{\varepsilon}_{RMS} = 100.4\%$), fixing the dynamics and nonlinearity did not show marked improvement over nonlinearities and dynamic estimates found
Figure 4-18: Muscle response estimates for high stimulation frequency input using fixed nonlinearity and fixed dynamics trained from 2Hz data.
iteratively ($\varepsilon_{RMS} = 49.6\%$). Figure 4-19 shows simulation errors for responses due to high frequency stimulation.

Figure 4-19: Simulations with fixed third-order polynomial nonlinearity and third-order dynamics

There was no consistent combination of nonlinearity and dynamics that produced the best fits. With fixed nonlinearities and/or dynamics, the quality of fits probably depended on the data set from which the nonlinearity and dynamics were obtained. If two data sets had output noises that correlated highly, fits would probably be better. However, if they had very different noise characteristics, the fits were probably very poor.
Iterative convergence and stability of the estimated dynamics were also issues for data sets with high stimulation frequency input. Of 150 simulations run on data sets with between 1 to 100Hz input stimulation frequency, 130 simulations (87%) converged. 114 (76%) resulted in stable estimated dynamic systems. However, for data sets with input stimulation frequency of 50Hz or more, 20 of 30 simulations (67%) converged, while only 11 cases (37%) were stable.

4.10 Comparison of all simulation results

After running ten model types on the fifteen sets of pilot data, the model performances were assessed. In this section, simulation errors and stability are compared across input stimulation frequencies and between linear and nonlinear models.

4.10.1 Simulation errors

Figure 4-20 compares all simulation errors, including those without the static input nonlinearity, those with an iteratively-found nonlinearity, and those with a fixed nonlinearity.

For stimulation frequencies of 20Hz and below, the RMS errors average 12%. Between 20 to 40Hz, simulations produced errors of 10%, and above 40Hz, errors average around 44%.

Table 4.1 shows, for each model, the RMS errors averaged across seven different frequencies. The averages are also plotted in Figure 4-22. The frequencies (1, 3, 5, 10, 15, and 20Hz) were selected on the basis that they converged and were stable for simulations of all ten model types. We see the average errors generated by all the model types are relatively consistent and generally fall within 10 to 14%; no model stands out in its ability to estimate contractile forces.
Figure 4-20: Summary of simulation errors (scatter and bar graphs)
Figure 4-21: Average simulation errors across stimulation frequencies

Figure 4-22: Average model errors across seven data sets below 25Hz
### Linear models (no nonlinearity)

<table>
<thead>
<tr>
<th>Experiment</th>
<th>Freq (Hz)</th>
<th>2nd ord dyn</th>
<th>3rd ord dyn</th>
<th>4th ord dyn</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID.001</td>
<td>1</td>
<td>7.94</td>
<td>6.21</td>
<td>5.86</td>
</tr>
<tr>
<td>ID.011</td>
<td>1</td>
<td>4.30</td>
<td>3.88</td>
<td>3.84</td>
</tr>
<tr>
<td>ID.003</td>
<td>3</td>
<td>17.97</td>
<td>16.73</td>
<td>14.96</td>
</tr>
<tr>
<td>ID.012</td>
<td>5</td>
<td>11.92</td>
<td>11.67</td>
<td>11.43</td>
</tr>
<tr>
<td>ID.005</td>
<td>10</td>
<td>12.63</td>
<td>10.85</td>
<td>10.33</td>
</tr>
<tr>
<td>ID.007</td>
<td>15</td>
<td>13.49</td>
<td>14.34</td>
<td>12.87</td>
</tr>
<tr>
<td>ID.002</td>
<td>20</td>
<td>26.87</td>
<td>27.45</td>
<td>12.56</td>
</tr>
</tbody>
</table>

Model average error

| 13.59 | 13.02 | 10.26 |

### Nonlinear models

<table>
<thead>
<tr>
<th>Experiment</th>
<th>Freq (Hz)</th>
<th>2nd ord dyn</th>
<th>3rd ord dyn</th>
<th>2nd ord dyn</th>
<th>3rd ord dyn</th>
<th>2nd ord dyn</th>
<th>3rd ord dyn</th>
<th>3rd ord nonlin, 2nd ord dyn</th>
<th>3rd ord nonlin, 3rd ord dyn</th>
<th>3rd ord fixed nonlin, 3rd ord dyn</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID.001</td>
<td>1</td>
<td>7.97</td>
<td>6.36</td>
<td>6.06</td>
<td>3.43</td>
<td>6.09</td>
<td>3.26</td>
<td>7.57</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ID.011</td>
<td>1</td>
<td>4.94</td>
<td>4.04</td>
<td>4.94</td>
<td>3.98</td>
<td>4.91</td>
<td>3.98</td>
<td>3.01</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ID.003</td>
<td>3</td>
<td>15.96</td>
<td>15.82</td>
<td>14.27</td>
<td>13.92</td>
<td>13.79</td>
<td>13.08</td>
<td>18.69</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ID.012</td>
<td>5</td>
<td>12.66</td>
<td>12.87</td>
<td>17.34</td>
<td>17.87</td>
<td>22.25</td>
<td>22.14</td>
<td>12.87</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ID.005</td>
<td>10</td>
<td>12.26</td>
<td>12.39</td>
<td>11.46</td>
<td>11.81</td>
<td>11.47</td>
<td>11.79</td>
<td>14.01</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ID.007</td>
<td>15</td>
<td>13.04</td>
<td>10.80</td>
<td>58.10</td>
<td>11.21</td>
<td>12.98</td>
<td>12.40</td>
<td>13.10</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ID.002</td>
<td>20</td>
<td>9.49</td>
<td>8.49</td>
<td>36.15</td>
<td>8.18</td>
<td>10.86</td>
<td>7.68</td>
<td>26.69</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Model avg error

| 10.90 | 10.11 | 21.19 | 10.06 | 11.75 | 10.62 | 13.70 |

Table 4.1: Average linear model RMS errors across seven data sets below 25Hz
The continuous-time system poles and zeros were also examined across all simulations. Since the discrete difference equation denominator coefficients had been relatively consistent across frequencies for second- and third-order dynamics, the continuous poles were expected to be relatively consistent as well. Figure 4-23 (also found in Appendix C) plots the poles and zeros for third-order dynamic systems. The real parts of the poles are relatively consistent across all data sets. The imaginary parts are non-zero. Since the estimates were not confined to the critically damped case, the poles were frequently estimated to be complex for both second- and third-order dynamics. As a result, oscillation could arise in the estimated force profile, yet this was not seen in the experimental evidence. When oscillatory behavior does arise in muscle, such as in patients with spasticity or tremor, it may be attributed to defects in the feedback sensing organs [52, 43]. However, this is clearly not the case for our healthy muscle. It is possible the complex poles may be attributed to noise and/or changes in the passive muscle force between stimulations: often, the baseline for inactive muscle fluctuates, as shown in Figure 4-24.

### 4.10.2 Simulation stability

Stability of the estimates was also compared across frequencies as well as between linear systems and nonlinear Hammerstein models. A model was ranked unstable (0) if it either had an infinite H2 norm, or the algorithm did not converge after over one hundred iterations. Typically, the algorithm took eleven to twenty iterations to converge. If the H2 norm was finite and the model converged, the model was ranked stable (1). The stability rankings across all ten (linear and nonlinear) simulations were then averaged for data at each stimulation frequency. As shown in Figure 4-25, stability of the estimated dynamics in general declined for frequencies above 40Hz (11 of 30 simulations), while data with stimulation frequencies of 20Hz and less generated more stable dynamic estimates (87 of 90 simulations). Between 20 to 40Hz, dynamic stability occurred in 53% (16 of 30) of the simulations.

To address differences in stability between linear dynamic estimates and Hammerstein dynamic estimates, a Student’s t-test was performed to compare the stability
Figure 4-23: Pole-zero maps for third-order dynamic systems with different input polynomial nonlinearities
Figure 4-24: Fluctuating passive muscle force: 1Hz data

Figure 4-25: Average stability of ten estimated models across stimulation frequency (1 = stable, 0 = unstable)
measures of the three linear versus seven nonlinear models. The standard deviation between the means for linear and nonlinear models was 12.2%, and the t-value was 0.47. To be statistically significant, the critical t for $p = 0.05$ is 2.05; therefore, the stability of the linear versus nonlinear models was not significantly different.
Chapter 5

Conclusion

For the pilot data of this thesis, the Hammerstein iterative identification algorithm seemed to be effective at identifying dynamically stable models (87 of 90 simulations) for muscle subjected to 20Hz or less pulse train stimulation frequency. The estimates generated an average RMS error of 12%. Between 20 to 40Hz, the simulation errors were of similar magnitude (10%), but the dynamic stability of the estimated models declined (16 of 30 simulations). Beyond 40Hz, the curve fits were very poor ($\bar{e}_{RMS} = 44\%$, 11 of 30 simulations dynamically stable); the information content of the data was too low to estimate the dynamics and/or recruitment curve adequately, and noise likely dominated the dynamic characterization.

Iterative convergence and stability were issues for data with high input stimulation frequencies. Of 150 simulations run on data with input frequencies ranging from 1 to 100Hz, 87% converged, often within the first eleven to twenty iterations, while the remaining 23% hadn’t converged by at least the 100th iteration. Of the 150 simulations, 76% were dynamically stable with finite H2 norms. When considering only the 30 simulations for data with input frequencies above 40Hz, the numbers drop: 67% converged, while 37% of the 30 simulations were dynamically stable.

Since data with stimulation frequencies of 20Hz and less generated reliable and stable estimates, the simulations run on these data sets allowed comparison between the different model types. Using a fixed nonlinearity produced comparable simulation errors as the bootstrapping procedure that simultaneously finds the recruitment
nonlinearity and the dynamics. This suggests that a straightforward estimate of the
dynamics in series with a known recruitment nonlinearity may be applied in lieu of
the potentially time-consuming bootstrapping procedure.

Likewise, linear dynamic and nonlinear Hammerstein models performed compara-
vably. This is possibly because the recruitment curves of the two types of models are
not necessarily so very different. Hammerstein models add a nonlinear recruitment
curve normalized to unity, while linear models assume a monotonically increasing
recruitment curve with no gain. However, if the zeros of the estimated dynamics are
appropriately manipulated, linear models can make up for changes in gain that would
otherwise be provided by Hammerstein recruitment nonlinearities.

However, by no means do these results confirm that the Hammerstein model
is inappropriate. Rather, this thesis argues that the Hammerstein muscle model
should be further explored to determine and understand what its limitations are.
Once it is determined at what frequency the Hammerstein model ceases to be an
effective description of isometric muscle behavior, the use and range of the model in
neuroprosthetic controllers may be better gauged.

5.1 Future work

Since this thesis conducted system identification on one group of pilot data sets, to
reach any firm conclusion would require additional testing and simulation. Moreover,
a few modifications to the model might be useful. For instance, constraining the non-
linearity to be monotonically increasing may help the algorithm find better solutions
[11, 8]. Adding another linear filter to precede the recruitment curve may also be a
useful way of better modeling calcium release [6] though it presents additional degrees
of freedom in the modeling process.
Appendix A

Experimental Protocol
**Purpose of Experiments**
Hammerstein system ID; see effect of stimulation frequency on recruitment curve

**Frog**  
*Rana pipiens* (female)  
Mass 85.96 grams

**Solution**  
Ringers  
No glucose  
Oxygenation Off  
Circulation Off

**Muscle**  
Plantaris longus (muscle belly only)  
Left Lo 36.05 (33.75 sut) mm  
Left mass 0.972 grams  
Right Lo 35.7 (syt) mm  
Right mass 0.986 grams

**Electrode**  
Suction  
Experiment duration 4sec isometric, 0 sec dynamic (unless otherwise noted)  
100 to 1000us (0.0001 to 0.001sec)

<table>
<thead>
<tr>
<th>Isometric Stimulation</th>
<th>Test ID</th>
<th>Bioreactor</th>
<th>Condition</th>
<th>Freq (Hz)</th>
<th>Force max (N)</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>LEFT MUSCLE</strong></td>
<td>ID.1</td>
<td>Secondary</td>
<td>10V, 1 pulse, random PW, no jitter</td>
<td>1</td>
<td>4.5N</td>
<td>exper duration too short (few data pts)</td>
</tr>
<tr>
<td></td>
<td>ID.2</td>
<td>Secondary</td>
<td>10V, 1 pulse, random PW, no jitter</td>
<td>20</td>
<td>8N</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ID.3</td>
<td>Secondary</td>
<td>10V, 1 pulse, random PW, no jitter</td>
<td>3</td>
<td>2N</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ID.4</td>
<td>Secondary</td>
<td>10V, 1 pulse, random PW, no jitter</td>
<td>50</td>
<td>6N</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ID.5</td>
<td>Secondary</td>
<td>10V, 1 pulse, random PW, no jitter</td>
<td>10</td>
<td>1.5N</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ID.6</td>
<td>Secondary</td>
<td>10V, 1 pulse, random PW, no jitter</td>
<td>75</td>
<td>6N</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ID.7</td>
<td>Secondary</td>
<td>10V, 1 pulse, random PW, no jitter</td>
<td>15</td>
<td>1.25N</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ID.8</td>
<td>Secondary</td>
<td>10V, 1 pulse, random PW, no jitter</td>
<td>100</td>
<td>5N</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ID.9</td>
<td>Secondary</td>
<td>10V, 1 pulse, random PW, no jitter</td>
<td>7</td>
<td>0.5N</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ID.10</td>
<td>Secondary</td>
<td>10V, 1 pulse, random PW, no jitter</td>
<td>30</td>
<td>2.5N</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ID.11</td>
<td>Secondary</td>
<td>10V, 1 pulse, random PW, no jitter</td>
<td>1</td>
<td>0.2N</td>
<td>12sec of stimulation</td>
</tr>
<tr>
<td></td>
<td>ID.12</td>
<td>Secondary</td>
<td>10V, 1 pulse, random PW, w/ jitter</td>
<td>5</td>
<td>0.5N</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ID.13</td>
<td>Secondary</td>
<td>10V, 1 pulse, random PW, w/ jitter</td>
<td>40</td>
<td>3.5N</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ID.14</td>
<td>Secondary</td>
<td>10V, 1 pulse, random PW, w/ jitter</td>
<td>25</td>
<td>1N</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ID.15</td>
<td>Secondary</td>
<td>10V, 1 pulse, random PW, w/ jitter</td>
<td>2</td>
<td>0.25N</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>max jitter 0.1791s</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>max jitter 0.0216s</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>max jitter 0.0351s</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>max jitter 0.4491s</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>RIGHT MUSCLE</strong></td>
<td>ID.16</td>
<td>Secondary</td>
<td>10V, 1 pulse, random PW, no jitter</td>
<td>2</td>
<td>3N</td>
<td>6sec of stimulation</td>
</tr>
<tr>
<td></td>
<td>ID.17</td>
<td>Secondary</td>
<td>10V, 1 pulse, random PW, no jitter</td>
<td>20</td>
<td>3N</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ID.18</td>
<td>Secondary</td>
<td>10V, 1 pulse, random PW, no jitter</td>
<td>10</td>
<td>1.25N</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ID.19</td>
<td>Secondary</td>
<td>10V, 1 pulse, random PW, no jitter</td>
<td>50</td>
<td>3N</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ID.20</td>
<td>Secondary</td>
<td>10V, 1 pulse, random PW, no jitter</td>
<td>3</td>
<td>0.75N</td>
<td>5sec of stimulation</td>
</tr>
<tr>
<td></td>
<td>ID.21</td>
<td>Secondary</td>
<td>10V, 1 pulse, random PW, no jitter</td>
<td>75</td>
<td>3.25N</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ID.22</td>
<td>Secondary</td>
<td>10V, 1 pulse, random PW, no jitter</td>
<td>15</td>
<td>0.75N</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ID.23</td>
<td>Secondary</td>
<td>10V, 1 pulse, random PW, no jitter</td>
<td>100</td>
<td>3N</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ID.24</td>
<td>Secondary</td>
<td>10V, 1 pulse, random PW, no jitter</td>
<td>7</td>
<td>0.4N</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ID.25</td>
<td>Secondary</td>
<td>10V, 1 pulse, random PW, no jitter</td>
<td>30</td>
<td>1.5N</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ID.26</td>
<td>Secondary</td>
<td>10V, 1 pulse, random PW, w/ jitter</td>
<td>5</td>
<td>0.1N</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ID.27</td>
<td>Secondary</td>
<td>10V, 1 pulse, random PW, w/ jitter</td>
<td>40</td>
<td>1.5N</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ID.28</td>
<td>Secondary</td>
<td>10V, 1 pulse, random PW, w/ jitter</td>
<td>25</td>
<td>0.5N</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ID.29</td>
<td>Secondary</td>
<td>10V, 1 pulse, random PW, w/ jitter</td>
<td>85</td>
<td>2.5N</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>max jitter 0.1791s</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>max jitter 0.0216s</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>max jitter 0.0351s</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**NOTES:**
* max jitter = 1/f - PW max so that pulse widths don't overlap; for safety, make that 0.9*(1/f - PW max)
* let stim frequency (1/pulse period) = 200Hz, train duration = 0.005sec so that pulse count = 1
* primary stimulator - very low voltages, so used secondary card for all experiments
* secondary stimulator - fuzz seen on oscilloscope as VCM servos to Lo, then diminishes as stimulation begins (cross-talk?)
* -15N voltage as baseline in data (though load cell reads 2.17N passive right)
Appendix B

MATLAB code
% dataFit.m
clear all
close all

file = 'experiments_22junO6/ID_015.sbr';
load(file,'-mat')
underscInd = find(file == '_'); file(underscInd) = '_';

time = FileData.TestData.Time;
forceUnfilt = FileData.TestData.Data(:,4);
    mean(FileData.TestData.Data(end-4000:end-1000,4));
posn = FileData.TestData.Data(:,1);

figure(1), plot(time,forceUnfilt), hold on
xlabel('time'), ylabel('force')
title (['measured data and corresponding stimulation: ', file])

wn = 100/(0.5*2000); %cutoff freq normalized to Nyquist freq = 1/2*Fs
[B,A] = butter(2,wn,'low','z'); %chose 2nd order Butterworth filter
force = filtfilt(B,A,forceUnfilt);
plot(time,force,'k')

% creating input
amplitude = str2num(FileData.StimulatorParameters.Amplitude);
pperiod = 1/str2num(FileData.StimulatorParameters.Frequency);
pwidth = str2num(FileData.StimulatorParameters.StimulationTime)*1e-6;
pcount = str2num(FileData.StimulatorParameters.TrainDuration)*...
    str2num(FileData.StimulatorParameters.Frequency);

whichStimPar = FileData.TestData.Data(1000,10); %detect varying param
if whichStimPar == 1 %voltage
    amplitude = FileData.TestData.Data(:,9);
end
if whichStimPar == 2 %pulse period
    pperiod = FileData.TestData.Data(:,9); %exclude intentional freq jitter
end
if whichStimPar == 3 %pulse width
    pwidth = FileData.TestData.Data(:,9); %is general smaller than 1 sample
end
if whichStimPar == 4 %pulse count
    pcount = FileData.TestData.Data(:,9);
end

vti=pwidth/1e6;
stimIndices = find(diff(FileData.TestData.Data(:,3)))+1;
u = zeros(size(vti));
u(stimIndices) = vti(stimIndices);

plot(time,u*5000,'r','LineWidth',2)
legend('measured\u2014force','LP\u2014filtered\u2014force','pulse\u2014width\u2014(ms)')

%to save memory/spare confusion, clear variables that won't be called again
clear FileData vti whichStimPar amplitude pperiod pwidth pcount

%%%%%%SETTING UP STRUCTURE%%%%%%%
basis = 'Chebyshev'; %options: 'RBF' 'Chebyshev' 'Legendre'

%%RBF fitting variables
numZeta = 4; %arbitrarily chosen spacing – can tweak
stdDev = 0.01; %arbitrary choice – can tweak

%%Polynomial fitting variables
polynOrder = 3; %highest polyn = P.polynOrder, # of polyn terms = polynOrder+1

%%Dynamic system variables
denOrder = 3; %number of system poles
numOrder = denOrder; %number of system zeros
delays = 5; %number of delays in discrete system

errorTrain(1:6) = [10000 1000 100 10 1 0.1]; %arbitrarily chosen errors
runNum = 1;
numIter = 50;

isomTrBounds = [6000 11000]; %6000:14000 for ID_011, 7000:11000 for ID_012
isomValBounds = [11001 14000]; %14001:30000 for ID_011

uTrain = u(isomTrBounds (1):isomTrBounds (2));
zTrain = force(isomTrBounds (1):isomTrBounds (2));
uValidate = u(isomValBounds (1):isomValBounds (2));
zValidate = force(isomValBounds (1):isomValBounds (2));

%%%%%%ISOMETRIC IDENTIFICATION – TRAINING DATA SET%%%%%%%
ulsomMin = min(min(uTrain),min(uValidate));
ulsomMax = max(max(uTrain),max(uValidate));
if strcmp(basis,'RBF') == 1
    [alnpTrain,isomTrCenters] = rbfSetup(uTrain,stdDev,numZeta);
else if strcmp(basis,'Chebyshev') == 1
    uTrainScaled = (uTrain–ulsomMin)/(ulsomMax–ulsomMin)*2–1;
    alnpTrain = chebyshevFit(uTrainScaled,polynOrder,1); %b/c scaled
else
    uTrainScaled = (uTrain–ulsomMin)/(ulsomMax–ulsomMin)*2–1;
    alnpTrain = legendreFit(uTrainScaled,polynOrder,1);
end
v100 = linspace(uIsomMin, uIsomMax)
if strcmp(basis, 'RBF') == 1
    aContCurve = rbfSetup(v100, stdDev, numZeta, isomTrCenters);
else if strcmp(basis, 'Chebyshev') == 1
    aContCurve = chebyshevFit(v100, polynOrder, 0);
else
    aContCurve = legendreFit(v100, polynOrder, 0);
end

% ESTIMATE V FIRST
vEstimTrain{1} = uTrain; %start w/ assumption v = u
h2norm(1) = 1000; %arbitrary choice
vError(1) = 1000; %arbitrary choice
alpha = 0.5; %0 if all new, 1 if all previous coeffs

cInpTrain{1} = pinv(aInpTrain)*vEstimTrain{1}; %coeffs of normalized linear IRC
vContCurve{1} = aContCurve*cInpTrain{1};
recrNorm = max(vContCurve{1});
cInpTrain{1} = cInpTrain{1}/recrNorm;
chybrid{1} = cInpTrain{1};
vContCurve{1} = aContCurve*chybrid{1};
vEstimTrain{1} = aInpTrain*chybrid{1};

while std(errorTrain(end-5:end))/errorTrain(end) > 0.05
    lti = iddata(zTrain, vEstimTrain{runNum}, 1/2000);
sysOE = oe(lti, [numOrder denOrder delays], 'Focus', 'Stability');
sysD = tf(sysOE.b, sysOE.f, 1/2000, 'variable', 'z');
    [a, b, c, d] = tf2ss(sysOE.b, sysOE.f);
sys = recrNorm*ss(a, b, c, d, 1/2000);
[impulseSave{runNum}, ImpSave{runNum}] = impulse(sys);
h2norm(runNum+1) = norm(sys);
for ind = 1:size(aInpTrain, 2)
    vEstimMx(:, ind) = lsim(sys, aInpTrain(:, ind));
end
cInpTrain{runNum+1} = pinv(vEstimMx)*zTrain;
chybrid{runNum+1} = chybrid{runNum}*alpha + cInpTrain{runNum+1}*(1-alpha);
vContCurve{runNum+1} = aContCurve*chybrid{runNum+1};
recrNorm = max(vContCurve{runNum+1});
chybrid{runNum+1} = chybrid{runNum+1}/recrNorm; %normalize nonlinearity
vEstimTrain{runNum+1} = aInpTrain*chybrid{runNum+1};
vContCurve{runNum+1} = aContCurve*chybrid{runNum+1};
[zEstimTrain, t, vStates] = lsim(sys, vEstimTrain{runNum+1}); %init cond later
errorTrain{runNum} = sum((zTrain-zEstimTrain).^2);
stability{runNum} = norm(tf(sysOE.b, sysOE.f, 1/2000, 'variable', 'z')) == Inf

102
runNum = runNum+1 
end 
runNum = runNum-1;

figure (3), plot(uTrain, vEstimTrain{runNum+1},'r'), hold on
figure (4), subplot(311), bode(sys), subplot(312), zplane(sysOE.b,sysOE.f)
subplot(313), impulse(sys)
figure, plot([1:1:runNum], errorTrain(1:end),'.'),
xlabel('run, number'), ylabel('error'),
title('force estimation squared error, u-training data')
figure, plot([1:1:runNum], stability ,'.'),
xlabel('run, number'), ylabel('stability'),
title('stability: yes, u=1, no, u=0')

[gainM, phaseM, wg, wp] = margin(sys);
disp('Gain margin'), disp(gainM)
disp('Phase margin'), disp(phaseM)
disp('Gain crossover frequency'), disp(wg)
disp('Phase crossover frequency'), disp(wp)

clear gainM phaseM wg wp

clear t tImpSave impulseSave

if strcmp(basis, 'RBF') == 1
    aInpValidate = rbfSetup(uValidate, stdDev, numZeta, isomTrCenters);
else if strcmp(basis, 'Chebyshev') == 1
    uValScaled = (uValidate-uIsomMin)/(uIsomMax-uIsomMin)*2-1;
    aInpValidate = chebyshevFit(uValScaled, polynOrder, 1);
else uValScaled = (uValidate-uIsomMin)/(uIsomMax-uIsomMin)*2-1;
    aInpValidate = legendreFit(uValScaled, polynOrder, 1);
end

vEstimValidate = aInpValidate*cHybrid{runNum+1};
zEstimValidate = lsim(sys, vEstimValidate,...
    time(isomValBounds(1):isomValBounds(2)), vStates(end,:));
errorValidate = sum((zValidate(2:end-1)-zEstimValidate(2:end-1)).^2);

figure (3), plot(uValidate, vEstimValidate,'r.'), plot(v100, vContCurve{runNum+1},'k'),
xlabel('u'), ylabel('estimated, V_u=f(u)'),
legend('training set', 'validation set', 'continuous estimate')

figure, plot(time(isomTrBounds(1):isomValBounds(2)), [zTrain; zValidate]), hold on
plot(time(isomTrBounds(1):isomTrBounds(2)), zEstimTrain,'r')
plot(time(isomValBounds(1):isomValBounds(2)),zEstimValidate,'g')
error = sqrt((errorTrain(end)+errorValidate)/(length(uTrain)+length(uValidate))
errStr = ['E_{rms}=';num2str(error)];
titleStr = ['Isometric_ID:',file];
delayStr = [num2str(delays),'delays'];
title({titleStr;errStr;delayStr});
xlabel('time'),ylabel('force_{isometric}')
indices = [];
for counter = 1:length(stimIndices)
    if (stimIndices(counter)>isomTrBounds(1))&&(stimIndices(counter)<isomValBounds(2))
        indices = cat(1,indices,stimIndices(counter));
    end
end
plot(time(indices),force(indices),'k.'),
legend('measured','training_fit','validation_fit','stimulation_points')
disp('rms_error:')
disp(sqrt((errorTrain(end)+errorValidate)/(length(uTrain)+length(uValidate))))
disp('max_force:')
disp(max(force(isomTrBounds(1):isomValBounds(2))))
disp('Stable_estimate_for_ID_00x?')
disp(stability(end))
disp('dynamic_system_numerator:')
disp(sysOE.b)
disp('dynamic_system_denominator:')

disp('zeros:')
disp(zeros');
disp('poles:')

charEqn = denom{1};
if denOrder == 2
    bnorm = charEqn(2); %this is b/m
    knorm = charEqn(3); %this is k/m (use measured mass to get k and b)
    disp('bnorm:'),disp(bnorm)
    disp('knorm:'),disp(knorm)
end
if denOrder == 3
    kbnorm = -poles(find(~imag(poles))); %this is k/b (for 3rd ord sys)
    complexPoles = poles(find(imag(poles)));
    bnorm = -2*real(complexPoles(1)); %this is b/m
    knorm = (abs(complexPoles(1)))^2; %this is k/m
    disp('kbnorm:'),disp(kbnorm)
    disp('bnorm:'),disp(bnorm)
    disp('knorm:'),disp(knorm)

104
end

% check for particularly big coeffs (bad)
figure, plot(cInpTrain), title('input\_nonlinearity\_coeffs')

% to check for frequency jitter (irregular) vs none (constant difference)
figure, plot(stimIndices(2:end) - stimIndices(1:end-1), ...), title('jitter\_check')

% clear file underscInd counter ind numIter indices %stimIndices
% clear aInpTrain aInpValidate dc lti vEstimMx
% clear v100 aContCurve ulsomMin ulsomMax uTrainScaled uValScaled
% clear isomTrBounds isomValBounds
% clear tImpSave impulseSave
% clear errTrStr errValStr titleStr
THIS PAGE INTENTIONALLY LEFT BLANK
Appendix C

Simulation Log
Figure C-1: No nonlinearity, second-order dynamics (Experiments 1 to 8: 1, 20, 3, 50, 10, 75, 15, 100Hz)
Figure C-2: No nonlinearity, second-order dynamics (Experiments 9 to 15: 7, 30, 1, 5, 40, 25, 2Hz)
Figure C-3: No nonlinearity, third-order dynamics (Experiments 1 to 8: 1, 20, 3, 50, 10, 75, 15, 100Hz)
Figure C-4: No nonlinearity, third-order dynamics (Experiments 9 to 15: 7, 30, 1, 5, 40, 25, 2Hz)
Figure C-5: No nonlinearity, fourth-order dynamics (Experiments 1 to 8: 1, 20, 3, 50, 10, 75, 15, 100Hz)
Figure C-6: No nonlinearity, fourth-order dynamics (Experiments 9 to 15: 7, 30, 1, 5, 40, 25, 2Hz)
Figure C-7: Difference equation denominator coefficient parameter variation
Figure C-8: Nonlinear recruitment curves from iterative identification with third-order dynamics (Experiments 1 to 9: 1, 20, 3, 10, 15, 7Hz)
Figure C-9: Nonlinear recruitment curves from iterative identification with third-order dynamics (Experiments 11 to 15: 1, 5, 40, 25, 2Hz)
Figure C-10: Simulations with fixed third-order polynomial nonlinearity and third-order dynamics (Experiments 1 to 6: 1, 20, 3, 50, 10, 75Hz)
Figure C-11: Simulations with fixed third-order polynomial nonlinearity and third-order dynamics (Experiments 7 to 12: 15, 100, 7, 30, 1, 5Hz)
Figure C-12: Simulations with fixed third-order polynomial nonlinearity and third-order dynamics (Experiments 13 to 15: 40, 25, 2Hz)
S-plane: 1st order nonlinearity, 2nd order dynamics

S-plane: no nonlinearity, 2nd order dynamics

S-plane: 2nd order nonlinearity, 2nd order dynamics

S-plane: 3rd order nonlinearity, 2nd order dynamics

Figure C-13: Pole-zero maps for second-order dynamic systems with different input polynomial nonlinearities
Figure C-14: Pole-zero maps for third-order dynamic systems with different input polynomial nonlinearities
Figure C-15: Pole-zero maps for fourth-order dynamic systems with different input polynomial nonlinearities
### Linear models (no nonlinearity)

<table>
<thead>
<tr>
<th>Experiment</th>
<th>Freq (Hz)</th>
<th>2nd ord dyn</th>
<th>3rd ord dyn</th>
<th>4th ord dyn</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID-001</td>
<td>1</td>
<td>7.938</td>
<td>6.2078</td>
<td>5.857</td>
</tr>
<tr>
<td>ID-011</td>
<td>1</td>
<td>4.301</td>
<td>3.879</td>
<td>3.842</td>
</tr>
<tr>
<td>ID-015</td>
<td>2</td>
<td>9.729</td>
<td>9.4006</td>
<td></td>
</tr>
<tr>
<td>ID-003</td>
<td>3</td>
<td>17.97</td>
<td>16.7266</td>
<td>14.9625</td>
</tr>
<tr>
<td>ID-012</td>
<td>5</td>
<td>11.92</td>
<td>11.6718</td>
<td>11.4281</td>
</tr>
<tr>
<td>ID-009</td>
<td>7</td>
<td>16.37</td>
<td>14.815</td>
<td>14.7137</td>
</tr>
<tr>
<td>ID-005</td>
<td>10</td>
<td>12.63</td>
<td>10.8458</td>
<td>10.3267</td>
</tr>
<tr>
<td>ID-007</td>
<td>15</td>
<td>13.49</td>
<td>14.3363</td>
<td>12.8699</td>
</tr>
<tr>
<td>ID-002</td>
<td>20</td>
<td>26.87</td>
<td>27.4536</td>
<td>12.5574</td>
</tr>
<tr>
<td>ID-014</td>
<td>25</td>
<td>14.31</td>
<td>14.1435</td>
<td>10.8565</td>
</tr>
<tr>
<td>ID-010</td>
<td>30</td>
<td>8.189</td>
<td>7.335</td>
<td></td>
</tr>
<tr>
<td>ID-013</td>
<td>40</td>
<td>10.92</td>
<td>8.5375</td>
<td>6.9236</td>
</tr>
<tr>
<td>ID-004</td>
<td>50</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ID-006</td>
<td>75</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ID-008</td>
<td>100</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Nonlinear models

<table>
<thead>
<tr>
<th>Experiment</th>
<th>Freq (Hz)</th>
<th>1st ord nonlin, 2nd ord dyn</th>
<th>1st ord nonlin, 3rd ord dyn</th>
<th>2nd ord nonlin, 2nd ord dyn</th>
<th>2nd ord nonlin, 3rd ord dyn</th>
<th>2nd ord nonlin, 3rd ord dyn</th>
<th>3rd ord fixed nonlin, 3rd ord dyn</th>
<th>Avg ( e_{RMS} ) across freq*</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID-001</td>
<td>1</td>
<td>7.976.606.60</td>
<td>6.303.433.26</td>
<td>3.26</td>
<td>7.57</td>
<td>6.07</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ID-011</td>
<td>1</td>
<td>4.944.944.91</td>
<td>4.943.983.98</td>
<td>3.98</td>
<td>3.01</td>
<td>4.18</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ID-015</td>
<td>2</td>
<td>10.7811.0912.78</td>
<td>10.7910.3112.99</td>
<td>10.57</td>
<td>10.52</td>
<td>10.53</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ID-003</td>
<td>3</td>
<td>15.9614.2713.79</td>
<td>15.8213.9213.08</td>
<td>13.08</td>
<td>13.09</td>
<td>13.02</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ID-009</td>
<td>7</td>
<td>15.48</td>
<td>13.8514.98</td>
<td>3.26</td>
<td>13.47</td>
<td>13.47</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ID-005</td>
<td>10</td>
<td>12.2611.4611.47</td>
<td>12.3911.8111.79</td>
<td>11.79</td>
<td>11.79</td>
<td>11.79</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ID-014</td>
<td>20</td>
<td>9.4936.1510.86</td>
<td>8.598.187.68</td>
<td>7.68</td>
<td>17.44</td>
<td>17.44</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ID-010</td>
<td>25</td>
<td>25</td>
<td>25</td>
<td>25</td>
<td>13.10</td>
<td>13.10</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ID-013</td>
<td>30</td>
<td>25</td>
<td>25</td>
<td>25</td>
<td>7.98</td>
<td>7.98</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ID-004</td>
<td>40</td>
<td>15.1312.61</td>
<td>10.9510.9510.95</td>
<td>13.75</td>
<td>10.78</td>
<td>10.78</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ID-006</td>
<td>50</td>
<td>43.95</td>
<td>33.78</td>
<td>33.78</td>
<td>33.78</td>
<td>33.78</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ID-008</td>
<td>75</td>
<td>5.88</td>
<td>5.88</td>
<td>5.88</td>
<td>5.88</td>
<td>5.88</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ID-008</td>
<td>100</td>
<td>76.41</td>
<td>64.71</td>
<td>64.71</td>
<td>64.71</td>
<td>64.71</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*average of RMS errors from both linear and nonlinear simulations (dynamically unstable and non-converging simulation errors denoted ‘—’)

**Average** \( e_{RMS} \)

- 12.43 between 1 and 20Hz, inclusive
- 10.36 between 25 and 40Hz, inclusive
- 43.57 between 50 and 100Hz, inclusive
- 16.34 fixed nonlinearity
- 16.62 iterated nonlinearity

Table C.1: RMS errors for all simulations
### Linear models (no nonlinearity)*

<table>
<thead>
<tr>
<th>Experiment</th>
<th>Freq (Hz)</th>
<th>2nd ord dyn</th>
<th>3rd ord dyn</th>
<th>4th ord dyn</th>
<th>Average stability</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID_001</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1.00</td>
</tr>
<tr>
<td>ID_004</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1.00</td>
</tr>
<tr>
<td>ID_007</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1.00</td>
</tr>
<tr>
<td>ID_015</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1.00</td>
</tr>
<tr>
<td>ID_003</td>
<td>5</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1.00</td>
</tr>
<tr>
<td>ID_007</td>
<td>7</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1.00</td>
</tr>
<tr>
<td>ID_005</td>
<td>10</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1.00</td>
</tr>
<tr>
<td>ID_002</td>
<td>15</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1.00</td>
</tr>
<tr>
<td>ID_009</td>
<td>20</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1.00</td>
</tr>
<tr>
<td>ID_001</td>
<td>25</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1.00</td>
</tr>
<tr>
<td>ID_003</td>
<td>30</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1.00</td>
</tr>
<tr>
<td>ID_002</td>
<td>35</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1.00</td>
</tr>
<tr>
<td>ID_004</td>
<td>40</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1.00</td>
</tr>
<tr>
<td>ID_006</td>
<td>50</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1.00</td>
</tr>
<tr>
<td>ID_008</td>
<td>60</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1.00</td>
</tr>
<tr>
<td>ID_007</td>
<td>70</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1.00</td>
</tr>
<tr>
<td>ID_008</td>
<td>75</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1.00</td>
</tr>
<tr>
<td>ID_005</td>
<td>80</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1.00</td>
</tr>
<tr>
<td>ID_009</td>
<td>90</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1.00</td>
</tr>
<tr>
<td>ID_007</td>
<td>100</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1.00</td>
</tr>
</tbody>
</table>

Average linear model stability: 0.8

### Nonlinear models

<table>
<thead>
<tr>
<th>Experiment</th>
<th>Freq (Hz)</th>
<th>1st ord nonlin, 2nd ord dyn</th>
<th>1st ord nonlin, 3rd ord dyn</th>
<th>2nd ord nonlin, 2nd ord dyn</th>
<th>2nd ord nonlin, 3rd ord dyn</th>
<th>2nd ord nonlin, 3rd ord dyn</th>
<th>3rd ord nonlin, 3rd ord dyn</th>
<th>3rd ord nonlin, 3rd ord dyn</th>
<th>3rd ord fixed nonlin</th>
<th>Avg nonlin stability</th>
<th>Avg overall stability**</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID_001</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1.00</td>
<td>1.00</td>
</tr>
<tr>
<td>ID_004</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1.00</td>
<td>1.00</td>
</tr>
<tr>
<td>ID_007</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1.00</td>
<td>1.00</td>
</tr>
<tr>
<td>ID_015</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1.00</td>
<td>1.00</td>
</tr>
<tr>
<td>ID_003</td>
<td>5</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1.00</td>
<td>1.00</td>
</tr>
<tr>
<td>ID_002</td>
<td>7</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1.00</td>
<td>1.00</td>
</tr>
<tr>
<td>ID_006</td>
<td>10</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1.00</td>
<td>1.00</td>
</tr>
<tr>
<td>ID_007</td>
<td>15</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1.00</td>
<td>1.00</td>
</tr>
<tr>
<td>ID_002</td>
<td>20</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1.00</td>
<td>1.00</td>
</tr>
<tr>
<td>ID_004</td>
<td>25</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1.00</td>
<td>1.00</td>
</tr>
<tr>
<td>ID_006</td>
<td>30</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1.00</td>
<td>1.00</td>
</tr>
<tr>
<td>ID_007</td>
<td>35</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1.00</td>
<td>1.00</td>
</tr>
<tr>
<td>ID_002</td>
<td>40</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1.00</td>
<td>1.00</td>
</tr>
<tr>
<td>ID_004</td>
<td>50</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1.00</td>
<td>1.00</td>
</tr>
<tr>
<td>ID_006</td>
<td>75</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1.00</td>
<td>1.00</td>
</tr>
<tr>
<td>ID_008</td>
<td>100</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1.00</td>
<td>1.00</td>
</tr>
</tbody>
</table>

Average nonlinear model stability: 0.74

* stable models denoted 1, unstable models denoted 0, non-converging models denoted —
** average stability of all ten linear and nonlinear models

Table C.2: Stability for all simulations
Bibliography


129


